DESIGN AUTOMATION FOR INTEGRATED OPTICS

by

Christopher Condrat

A dissertation submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Electrical and Computer Engineering

The University of Utah

August 2014

The University of Utah Graduate School

STATEMENT OF DISSERTATION APPROVAL

The dissertation of **Christopher Condrat**

has been approved by the following supervisory committee members:

Priyank Kalla	, Chair	**12 / 18 / 2013** Date Approved
Steven Blair	, Member	**12 / 18 / 2013** Date Approved
Chris Myers	, Member	**12 / 18 / 2013** Date Approved
Kenneth Stevens	, Member	**12 / 18 / 2013** Date Approved
Erik Brunvand	, Member	**12 / 18 / 2013** Date Approved

and by **Gianluca Lazzi** , Chair/Dean of

the Department/College/School of **Electrical and Computer Engineering**

and by David B. Kieda, Dean of The Graduate School.

ABSTRACT

Recent breakthroughs in silicon photonics technology are enabling the integration of optical devices into silicon-based semiconductor processes. Photonics technology enables high-speed, high-bandwidth, and high-fidelity communications on the chip-scale—an important development in an increasingly communications-oriented semiconductor world. Significant developments in silicon photonic manufacturing and integration are also enabling investigations into applications beyond that of traditional telecom: sensing, filtering, signal processing, quantum technology—and even optical computing. In effect, we are now seeing a *convergence of communications and computation,* where the traditional roles of optics and microelectronics are becoming blurred.

As the applications for opto-electronic integrated circuits (OEICs) are developed, and manufacturing capabilities expand, design support is necessary to fully exploit the potential of this optics technology. Such design support for moving beyond custom-design to automated synthesis and optimization is not well developed. Scalability requires abstractions, which in turn enables and requires the use of optimization algorithms and design methodology flows. Design automation represents an opportunity to take OEIC design to a larger scale, facilitating design-space exploration, and laying the foundation for current and future optical applications—thus fully realizing the potential of this technology.

This dissertation proposes design automation for integrated optic system design. Using a building-block model for optical devices, we provide an EDA-inspired design flow and methodologies for optical design automation. Underlying these flows and methodologies are new supporting techniques in behavioral and physical synthesis, as well as device-resynthesis techniques for thermal-aware system integration. We also provide modeling for optical devices and determine optimization and constraint parameters that guide the automation techniques.

Our techniques and methodologies are then applied to the design and optimization of optical circuits and devices. Experimental results are analyzed to evaluate their efficacy. We conclude with discussions on the contributions and limitations of the approaches in the context of optical design automation, and describe the tremendous opportunities for future research in design automation for integrated optics.

This work is dedicated to my parents.

CONTENTS

ACKNOWLEDGEMENTS

First and foremost, I'd like to thank my parents for their love and support throughout the entire process of obtaining this PhD and everything before it. It has been a *very* long journey. Never once have they questioned or discouraged my dedication to this endeavor—but quite the opposite. Without them, this work would not have been possible, and I've been truly blessed in this regard. I hope I make them proud.

Second, I'd like to thank my advisor for my graduate career: Priyank Kalla. It has been a pleasure working under his guidance at both a professional and personal level. Priyank's knowledge, openness, flexibility, and patience have facilitated my growth as a researcher, scientist, and writer. I appreciate that I can converse and banter with Priyank as an equal, that he is willing to sacrifice his own sleep to help me finish a paper, and that he motivates me when I'm feeling discouraged. My time as a graduate student has been a great experience.

I also extend my thanks to my PhD committee members—Chris Myers, Steve Blair, Ken Stevens, and Erik Brunvand—who have taken time to engage in this research and have helped me in my education throughout my career. Steve Blair, especially, has been a vital part of this work: our connection to the world of integrated optics. Thanks also to my friends in the ECE office staff. In addition to your warmth, you've always ensured that I haven't suffered any hiccups throughout the years.

Thanks to my sister, Michelle, for being such a cool sibbling when growing up and still as an adult. Only after experiencing my friends' sibblings did I realize how lucky I was and am to be friends with you.

Thanks also to my grandparents, uncles, aunts, and cousins who have taken interest in my studies and are proud of what I've achieved. I appreciate your patience when I try to explain what I'm researching in grounded terms.

I'd like to thank Dante and Karen Bertelli, two of the best friends I've had the opportunity to know. Dante is one of the few persons I can converse with, eye-to-eye, on virtually all technical subjects. Our times working, engineering, camping, partying, brainstorming, and even working on school projects have always been a blast. We're truly on the same wavelength.

I'd like to thank the other members of G6 Networks, the company we founded in 2000 after high school. Thanks to Joseph and Wendy Harkins for your friendship and the great years working in G6, even if I haven't seen you for a while. Thank you Will Aoki for your friendship, hilarious writing, and help with keeping G6 computer hardware functioning—and even when my parents needed help. Thanks also to the former members of New Moon Media for taking our scrappy company under its wing and exposing us to a professional branding and design firm. Chris Künstadt is not only an excellent designer and media director, but a good friend whose humor made the office feel so welcoming. I grew a lot as a developer, animator, and designer working with Chris. Jamie too: it was always hilarious to make you laugh with the crazy stuff we'd find on the net.

I'd like to thank Vinh Pham, for the years of friendship, the experiences we've had, his willingness to take me outside my comfort zones, and the friends I've met through him. It was a great fortune to have the same electrical engineering lab that connected us a decade ago. Vinh is a gateway friend—through whom I met Matt Schultz, Duc Phuong, and Chau Nguyen. Vinh, Matt, and Duc are exceptional friends, and our week-long visits to "That Thing In The Desert" have demonstrated that time and time again. Chau, I always value your advice and wisdom, and I hope to learn more from you.

Thanks to Sivaram Gopalakrishnan for your friendship during our years in the lab and also your help during my interview process. Your insight into many things was enlightening, even if you didn't realize it, and we had a lot of great laughs in the office. Vijay Durairaj and Namrata Shekhar were also great lab buddies (even if I was rarely there to enjoy your company). I can't forget Larry Schlitt either: you're a smart guy, and you need to continue this work. Thanks Xiaojun Sun for helping me get this to the thesis office as well!

For everyone else that wasn't specifically mentioned, you're not forgotten, but the stage is lowering, and I need to email this off to my committee members before it is too late. Thank you!

CHAPTER 1

INTRODUCTION

Advancements in integrated optics are expanding the role of optical devices in system design. Opto-electronic integrated circuits (OEICs) [1], merging optics and control electronics on a monolithic substrate, are now a reality and enable optical integration in a diverse set of applications, such as sensing, signal processing, communications, and also computing [2]–[9]. The driving forces behind optics technology comes from different, but interrelated areas. One area is *optical interconnects.* As semiconductors feature sizes have scaled downward, metal interconnects are now the dominant cause of delay and power usage in system design. In addition, the trend towards greater parallelism at the system level [10] has prioritized the role of communications in computing. Optics are therefore being pushed as an inter- and intrachip *interconnect technology* to provide high-speed, long-haul, *low-power* communications [11]–[16]. The ability for optics technology to fulfill this communications role hinges on the ability to deploy optics at the endpoints of data transmission and throughout the substrate.

A second driving force behind optical technology is therefore that of manufacturing. Significant developments in microphotonics have come from the ability to manufacture *silicon-based* optical devices. Traditionally, the separation between optics and microelectronics has been one of process differences: whereas microelectronics can utilize silicon as a semiconductor, optical devices have traditionally relied on more group III–V semiconductors to create usable optical devices (lasers [17], [18], high-speed modulators [19], [20], detectors [21]). The use of silicon enables integrated optics to leverage mature silicon-based semiconductor processes as well as enable such optical devices to integrate directly in system designs. The lack of process compatibility has stymied cross-domain integration until recently. We are now in a position to fully realize the potential of the marriage of optics and microelectronics.

Moving beyond optics as a complementary communications technology is the push for optics as a *computing technology* [2]–[9]. Optics has always been considered the next step in computing technology; however, the great success of complementary metal-oxide-semiconductors (CMOS) in silicon-based processes, coupled with the cost and a lack of large-scale optical manufacturing

capability, have stymied the development of high density optical computing systems. Recent advances in optical system manufacturability and technology support presents new opportunities in design-space exploration. Initiatives such as Optoelectronic Systems In Silicon (OpSIS) [22]—the optoelectronic counterpart to Metal Oxide Semiconductor Implementation Service (MOSIS) [23]—are enabling researchers the ability to fabricate integrated optical designs within relatively inexpensive, but cutting-edge silicon optical processes. Support from industry [24], [25], government [26], and academia [22] are also pushing such research in order to develop optical technologies that extend beyond the traditional roles of telecommunications.

We are now reaching a threshold where photonic integration is possible beyond the traditional limits of telecom technology, driven by interdisciplinary research and development to fully utilize this technology. The need is here, but the necessary design support to take optical design beyond that of manual and hand design is not well developed. As optical devices are integrated on larger scales, the need for design automation becomes apparent to handle greater levels of complexity in design. Scalability requires abstractions, which in turn enables and requires the use of optimization algorithms and design methodology flows. Design methodology flows are key to partitioning large systems into realizable subcomponents, which may be characterized and optimized at different levels of abstraction. This process enables synthesis and optimization techniques to be further refined and expanded within an automation framework for performance improvement and reliability.

The feasibility of this approach has already been demonstrated to great success for microelectronics. There is interest in replicating this success in optical design and integration. This thesis takes steps in this direction, proposing a design automation flow with abstractions, optimization algorithms, tool-flows, and methodologies—enabling the synthesis of OEICs through automated means. Design automation represents an opportunity to take OEIC design to a larger scale, facilitating design-space exploration and laying the foundation for current and future optical applications—fully realizing the potential of this technology.

1.1 Developments in CMOS and Microphotonics

Silicon is the mainstay of the semiconductor industry. The ease of manufacturing for semiconductors in well-characterized silicon-based processes and steady improvements in performance and density at each process node makes CMOS-based technology the dominant computing manufacturing technology. For the same reasons, attempts were also made to develop silicon-based integrated optics—*silicon photonics*. Early attempts, however, proved fruitless, except for passive waveguide devices.

Silicon's indirect band gap means that silicon cannot produce light based on classic electron transitions. All-silicon lasers were considered all but impossible until 2004 with the development

of [27] and in 2005 with Intel's all-silicon, continuous-wave Raman laser [28]. These lasers utilize stimulated Raman scattering rather than electron transitions to produce light emission; however, such lasers are very large and experimental, and miniaturization efforts are still underway. Silicon's band gap also prevents it from detecting light at normal telecom wavelengths, requiring materials—such as germanium—for light detection. Incorporating a material such as germanium into silicon-based processes is difficult because of lattice incompatibilities. Without the necessary additional device support in modulation and light emission, hybrid processes were not pursued.

Despite silicon's limitations in both light emission and detection, the major hurdle in silicon-based optics has always been *modulation.* The electro-optic effects in silicon are either absent, in the case of Pockels effect [29], or very weak in the case the Kerr effect [30]—traditionally limiting optical modulation to practically useless tens of Mhz. Thermo-optic modulation is also very slow and power hungry. Therefore, in order to achieve necessary data rates (> 1 Ghz, but ideally 10s to 100s of GHz), III–V semiconductor compounds such as gallium arsenide (GaAs) and indium phosphide (InP), or materials such as lithium niobate ($LiNbO_3$) would become the materials of choice for photonic modulators. The ability of III–V semiconductors to realize functioning laser devices also makes them attractive for manufacturing optical systems. Integrated optics would therefore be limited to nominally telecom applications—where the long-distance, high-fidelity of optical communications coupled with necessary high data-rates justified the expense of specialized manufacturing processes.

Silicon would remain the subject of photonic research throughout the 1980s and 1990s. In particular, all-silicon passive waveguide devices, such as Array Waveguide Gratings (AWGs) [31] used for multiplexing, remained a useful application of so-called *1st-generation* silicon photonics [32], [33]. In addition, a number of useful results would also be discovered in the 1990s, notably the characterization connecting refractive index changes to free-carrier concentration in silicon by [34]. This important result would be the breakthrough that propelled *2nd-generation* silicon photonics a decade later.

In 2005, Intel Corporation announced the first all-silicon optical modulator operating beyond the 1 Ghz threshold [19]. In contrast to modulators based on electro-optic modulation, where an applied electric field directly modulates light, Intel's modulator relies on phase changes induced by refractive index changes due to carrier concentration. Within a Mach–Zehnder device topology, this causes constructive and destructive interference at the output, enabling light modulation. Though relatively slow compared to contemporary modulators, Intel's modulator represented a significant breakthrough in silicon photonics. An all-silicon optical modulator at usable modulation speeds (> 1 Ghz) meant that viable optical networks could be fabricated in

all-silicon processes. This development ushered in a number of subsequent breakthroughs in silicon photonic device development, including faster modulators [20], [35], hybrid lasers [17], and other device technologies [36]. The viability of all-silicon optical modulation also prompted the development of hybrid silicon-germanium processes [37] incorporating strained silicon techniques to enable germanium bonding to silicon materials. In conjunction with optical modulators, such processes enable fully functioning optical systems—including light sources from external lasers coupled to the system [38].

The promise of monolithic integration of photonic networks in silicon-based processes opens the door to a great number of opportunities in system design. Optics, once an exclusively telecom technology, is now able to leverage advanced processes in fabrication and integration. This change enables far greater flexibility and complexity in design as well as the ability for designers to investigate novel methods for utilizing optics in systems. Already a number of architectures have been proposed for connecting systems via optical interconnect networks [14], [39], including as separate layers in 3D integrated chips (ICs). Investigations have also been made into optical digital signal processing [40], sensing, and even computing frameworks that can leverage optics in ways that would have been cost prohibitive. In essence, we are now seeing a *convergence of computation and communications.*

What is now lacking is the design automation infrastructure to design and build optical system using more advanced methodologies. The expense and specialization of integrated optics systems has traditionally meant that there was little incentive to invest in automated means to construct networks of devices. With expanded device integration, electronic design automation (EDA)-like optical design flow methodologies are necessary to enable design beyond the small scale. Optical toolsets such as those from Lumerical [41], RSoft [42], and others are suitable for the design and analysis of smaller optical systems, but are not designed with automation in mind, nor do they provide the necessary abstractions to scale into larger systems. By abstracting the optical design process by using concepts such as logical building-block models, we can exploit decades worth of EDA techniques and automation design patterns to automate the process of optical design.

Optical switching is particularly well suited to abstraction. By treating optical switching devices as digital switching elements, we can model optical routing as a network of building-blocks connected by waveguides. Modeling these optical building blocks within a Boolean logic framework enables us to applying logic optimization techniques in the same manner as those applied to EDA netlists. Physical synthesis is also well adapted to automation, as placement and routing techniques utilized in EDA are equally applicable to optical devices—while still requiring technology-specific constraints and objectives within the optimization engines.

1.2 Electronic Design Automation

It is customary to describe design automation techniques in the context of a design flow. Figure 1.1 depicts an architypical EDA design flow we adapt for optical design automation. The design flow divided into two major phases: 1) *behavioral synthesis,* where a circuit specification is transformed, optimized, and mapped onto a library of technology-abstracted, interconnected building blocks forming a *netlist* and 2) *physical synthesis,* where the netlist is transformed into a physical layout through device placement and the routing of wires, power rails, clocks, and other interconnects to and between such devices. The result of this design flow, after numerous rounds of verification, validation, and sign-off, is a layout ready for lithographic manufacturing.

The design flow only captures the overall structure of synthesis. The details are important, especially in how we bridge the gap between technology and abstraction. More specifically, we extract technology-specific parameters to constrain the design process. Understanding how transistor-based parameters apply to EDA is key to understanding how we may also model a design flow for optics.

1.2.1 Behavioral Synthesis

Behavioral synthesis is a key step in the process of system design and relies on the idea that technology elements can be modeled as (logical) building-blocks. A building-block model enables networks of devices to be analyzed and optimized at different levels of abstraction. In high-level (electronic system level) synthesis (HLS), designs are decoupled from the underlying hardware entirely in order to concentrate on its architectural, algorithmic, and resource design parameters. Behavior and algorithmic level languages such as SystemC are transformed and optimized into Register-Transistor Level (RTL) descriptions with an emphasis on resource binding and allocation, scheduling, and constraint generation. Ensuring that a design is testable is also an important consideration at this stage, ensuring that correctness can be accounted for at lower levels of abstraction.

The output of HLS is an RTL description to be mapped to the underlying hardware building blocks. Interconnected devices are decomposed into subsets using partitioning algorithms. These subcircuits are then transformed and optimized leveraging powerful concepts such as Boolean logic, functional representation including equation forms, and graph representations such as And-Inverter Graphs (AIGs) [43] and Binary Decision Diagrams (BDDs) [44], optimization methodologies, and composition/decomposition techniques to design and optimize circuits. The result is a netlist ready for physical synthesis.

In semiconductor technology, the *transistor* serves as the basis element for constructing logical building blocks. Transistors are connected together in design patterns such as static-CMOS, to form

Boolean logic gates and other elements (flip-flops, etc.) to provide the necessary device abstraction for synthesis. Logics formed from transistors are especially appealing in that they can switch for large signal gain and therefore provide signal restoration at the output of gates or logic networks. The resulting effect is an extremely versatile logic composition paradigm, enabling function output sharing (fan-out), cascading of gates, and signal isolation.

Modeling building blocks is important for optimization and operational correctness. At lower levels, logic devices are modeled and analyzed using extracted Simulation Program with Integrated Circuit Emphasis (SPICE) models, which capture a device's many properties, nominally the device's effect on its input and output signals (e.g., AC/DC characteristics, noise, impedance, etc.). Such analysis is resource intensive. Therefore, device networks are first analyzed using first-order approximation models such as the logical effort of gates, derived from a gate's topology and size/capacitive load ratio, and parasitics. These models are useful for fast, first-order approximations used in timing optimization.

Optimization parameters for behavioral synthesis include timing, power, and area, but also testability and accounting for effects analyzed during physical synthesis. Timing affects both performance and correct operation and is derived from the delay characteristics of the technology's circuit elements (e.g., logical effort) and their interconnects (fan-outs, interconnect capacitance, etc.). Area is usually measured in terms of numbers of gates, the size of such gates, and the necessary interconnects to connect such gates. Power is also an important consideration, especially as power is no longer simply measured in terms of the switching activity of a circuit, but also in terms of leakage. Power management is an important part of behavioral synthesis and also affects timing and area through additional circuitry and from the side-effects of power-saving operations. Designing for testability is also a necessary component in behavioral synthesis, ensuring that defects in manufacturing are detected before shipment. Incorporating testing structures such as scan chains into the design improves coverage and reduces the complexity of test-pattern generation.

1.2.2 Physical Synthesis

The output of the behavioral synthesis phase is a *netlist* comprising a set of interconnected logical building blocks. These devices must be assigned physical locations in the manufacturing substrate, and the devices must be routed together with "wires" or their equivalent. Physical synthesis is broken down into two subphases that are often interrelated: placement and routing [45]. Additional effects such as hotspot detection in chip planning are also important considerations to improve routability, power consumption, heating, and yield.

Considering devices as geometric blocks connected by wires enables us to model the placement problem as that of arranging blocks within the substrate. Most designs have some sort of hierarchical

structure made up of modules, and the placement often reflects this structure. At all levels, a number of metrics guide the procedure such as: area, wire-length between connected devices, cuts across routing regions, signal delay, potential wire congestion that the routing phase may encounter, and also thermal management. At the lowest levels, placement involves packing individual devices together into compact structures such as standard cell rows. Techniques such as min-cut placement, quadratic (2-D) placement, force-directed placement, and simulated annealing are common techniques for placement.

Global routing is an important part of physical synthesis, ensuring that nets are routed as rectilinear Minimum Steiner Trees. Wire-length has traditionally dominated the guiding metrics for placement; however, this has changed with congestion becoming important for improving the subsequent routing phase. Global routers such as [46] use mixed-integer linear programming (MILP) methods and maze routing formulations to reduce congestion and wire-length. As the placement of devices directly affects the routing of such devices, global routing is often bound to the placement phase. Techniques such as [47] integrate placement and routing, ensuring that any placement of devices will also ensure that routing regions have sufficient capacity to connect devices and modules together.

With placement and global routing complete, detailed-level routing is performed. This stage is responsible for routing the individual wires to their local destinations. Routing regions are often broken down into 1) channels, where pins are found on only two sides of the region, and 2) switchboxes, where pins may be found on any side. Various techniques for assigning wires to metal layers are used [48]–[50] in order to produce a routing solution. In some cases routing may occur over logic cells to complete, should there be enough room.

1.3 Proposed Design Flow

The design flow proposed in this dissertation, depicted in Fig. 1.2, draws inspirations from EDA design flows and methodologies. As in the previously described EDA design flow, this optical design automation flow is divided into two major phases: behavioral synthesis and physical synthesis. Ancillary to this flow is *technology modeling,* where the groundwork is laid for design automation in terms of building-blocks models and optimization metrics used throughout the design flow. System integration is also important and role by introduces external constraints and effects on the optical system such as area-limitations, packaging, and thermal interactions between on-chip heat sources and optical devices.

Our contributions to this design flow are indicated in Fig. 1.2. For the behavioral synthesis phase, we model optical switching elements as building-blocks that implement Boolean logic functions. As a communications technology, this building block model is derived from conventional electro-

optical routing devices. By using conventional routing devices, synthesis techniques will remain applicable to communications networks as well as frameworks that may utilize optical logic. As in EDA behavioral synthesis, this abstraction also enables us to utilize Boolean logic concepts and techniques to design and optimize optical networks, and leverage Boolean logic synthesis techniques developed for EDA—suitably refined and extended within the constraints of the technology.

With a building block model complete, our task is now to develop optics-specific logic synthesis techniques to construct and optimize *optical netlists*. These netlists are composed of optical switching elements and waveguide interconnects and form the input to the physical synthesis stage.

The physical synthesis stage comprises the placement of optical devices and the routing of those devices using waveguides. The building-blocks used in behavioral synthesis are also used for physical synthesis, treated as logic cells for placement, with waveguides acting as interconnecting wires. Much like the device placement strategies employed for microelectronic designs, area is an important consideration. Wire length, to a first degree, can also aid in ensuring routability. An important modeling constraint we impose on the routing problem is that waveguides are fabricated in *single-layer* planar substrates. While this still allows for waveguides to cross—perpendicular to each other—and retain their signals, this comes at with a signal loss penalty.

Signal loss is identified as the major guiding metric in optical physical synthesis. Integrated optics lacks the signal restoration properties of static-CMOS and therefore signal losses must be minimized across designs. All devices in an optical network have insertion losses, and even the interconnecting waveguides can affect signal integrity. The aforementioned waveguide crossings in particular are important loss sources, and as networks scale in complexity, the need to minimize waveguide crossings is critical. The techniques incorporated into our design flow—at all stages—are therefore centered around reducing signal loss.

Thermal considerations are also an important part of this optical design flow. For CMOS-based technologies, thermal management is an important part in ensuring correct operation of designs; rises in temperature can cause delay faults, premature aging, and also cracking of chips. In the optics domain, however, we are more concerned with the thermal *stability* of optical devices. The properties of optical materials, notably the refractive index, are affected by temperature. This effect is often small and negligible, but certain sensitive devices are exceptionally affected. We, therefore, also include a thermal-aware resynthesis step for photonic switching devices as a part of the design flow.

1.3.1 Thermal-Aware Resynthesis

The need for fast, compact, optical modulators and routing devices has led to the development of resonant devices such as optical ring resonators that can deliver high performance in a small

footprint. The resonant operation of the ring resonator enables small variations in refractive index to effect large changes in wavelength response. This effect also makes ring resonators sensitive to temperature changes that change the refractive index of the medium.

Integrating such devices into the network fabric of systems will ultimately require tuning to compensate not only for process variation, but the effects of temperature gradients in the substrate. Such tuning is power intensive, requiring per-device microheaters and feedback circuitry to actively compensate for thermal variations or offset voltages applied to P-i-N junctions on the ring. If thermal gradients can be estimated ahead of time, compensation methods can be applied as permanent modifications to temperature-sensitive devices such as ring resonators—complementing active tuning techniques and saving power.

1.4 Contributions of This Dissertation

This dissertation proposes *design automation for integrated optic system design.* Drawing inspirations from EDA design flows, we develop a photonic system design flow for automating network design and physical synthesis. This automation design flow changes how we approach integrated optic design, enabling the move beyond traditional manual design and layout to that of automated techniques. As the scale of device integration grows, automation techniques will be critical in order to fully realize the potential of the technology.

We model device abstractions that capture key parameters of optical switching devices that enable a building-block composition methodology. This building-block methodology is used both for behavioral synthesis and physical synthesis. We also identify optimization objectives for guiding automation techniques:

- **Behavioral Synthesis:** Optical devices are modeled as logic elements to employ powerful concepts from Boolean logic theory for functional composition and optimization [2], [3]. Each optical gate incurs insertion losses. We therefore develop technology-specific common subexpression-based techniques to reduce gate counts [2].
- **Physical Synthesis:** Optical devices are placed in a standard-cell-style layout topology, where the device-blocks with ports are connected by waveguide interconnects. Waveguide crossings and bends act as signal loss mechanisms and are optimized for in both global and detailed routing. [51], [52]
- **Thermal-Aware Device Resynthesis:** Thermal interactions between on-chip heat sources and ring resonators pose significant operational challenges. To account for external thermal gradients, we present resynthesis templates that exploit waveguide geometry for temperature compensation of optical devices. We present a methodology for constructing ring-resonator

templates enabling process-compatible compensation, incorporating temperature-range and precision parameters [53] (submitted and in review).

This work also demonstrates how contemporary EDA design techniques and methodologies are adapted for optical design:

- **Boolean Decomposition:** New technology-specific XOR-based Boolean decomposition techniques are applied to logic synthesis for integrated optics [2].
- **Optical Device Placement:** EDA partitioning and placement techniques are applied to optical device placement. Devices are placed into rows suitable for channel-based routing [54].
- **Global Routing:** A mixed integer linear programming (MILP) problem formulation and methodology is presented for signal-loss-constrained routing [54].
- **Detailed Routing:** We suitably adapt and constrain conventional channel routing techniques to minimize waveguide crossings. New channel-routing techniques are developed [51], [52] ([55] in review); these techniques are also applicable to general-purpose channel routing.
- **Device Layout:** Using contemporary Very Large-Scale Integration (VLSI) layout tools, we construct a system layout for an optical design for fabrication using predesigned optical devices as interconnected building blocks. This design was fabricated in an optical process [22] and is depicted in Fig. 1.3.

1.4.1 Thesis Organization

The remainder of the dissertation is organized as follows. Chapter 2 reviews integrated optics technology and theory, providing an overview of integrated optic systems in the context of this work. Subsequent chapters cover the various topics that form the main body of research in this dissertation. As the breadth of design automation is large, we address previous work and relevant preliminaries within the individual subsections of this dissertation.

- Chapter 3 covers behavioral synthesis, where we describe our logical building-block model, our Boolean logic composition methodology, and a logic synthesis technique incorporating common subexpression extraction to reduce gate count.
- Chapter 4 presents our overall physical synthesis methodology. We also describe design constraints for physical synthesis, metrics, and routing grid modeling.
- Chapter 5 presents a global-routing formulation for optical waveguide routing. This is formulated within a crossing-constrained mixed-integer linear programming (MILP) problem methodology.
- Chapter 6 covers our detailed routing approach, specifically channel routing for optical

waveguides. We present two channel routing techniques and show how each reduces signal losses by minimizing waveguide crossings and bends through routing constraints.

- Chapter 7 presents our methodology for thermal-aware resynthesis for photonic ring-resonator devices. Ring resonators are key components of many optical network architectures; however, their temperature sensitivity presents challenges to integration. We analyze the effects of thermal changes on ring-resonators and present a template-based, ring-resonator design enabling process-compatible resynthesis for thermal compensation.
- Chapter 8 concludes this dissertation, reflecting on the research performed as well as future research directions in this area.

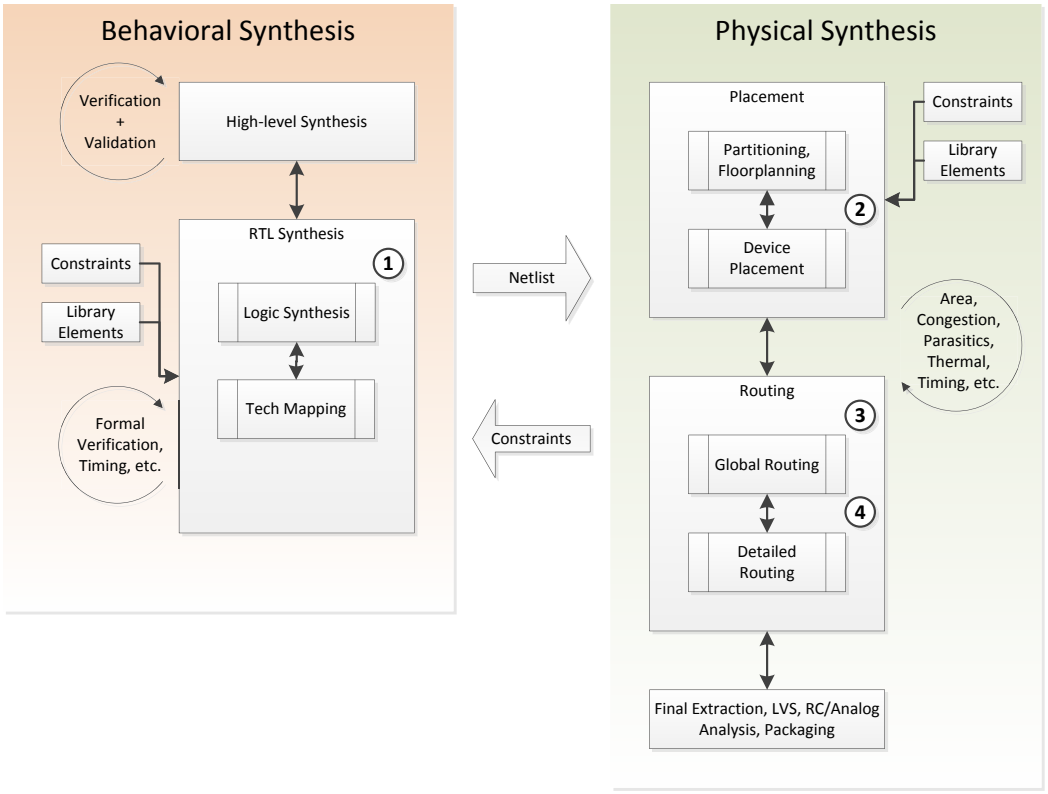


Fig. 1.1 Electronic design automation (EDA) flow

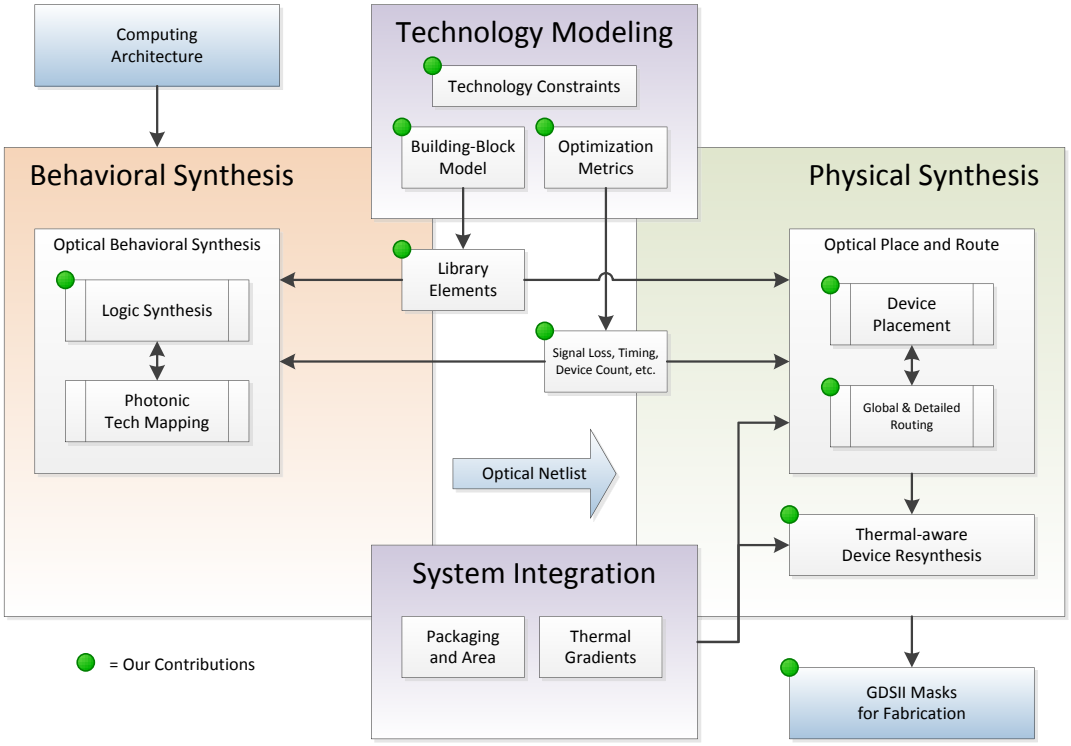


Fig. 1.2 The proposed design flow

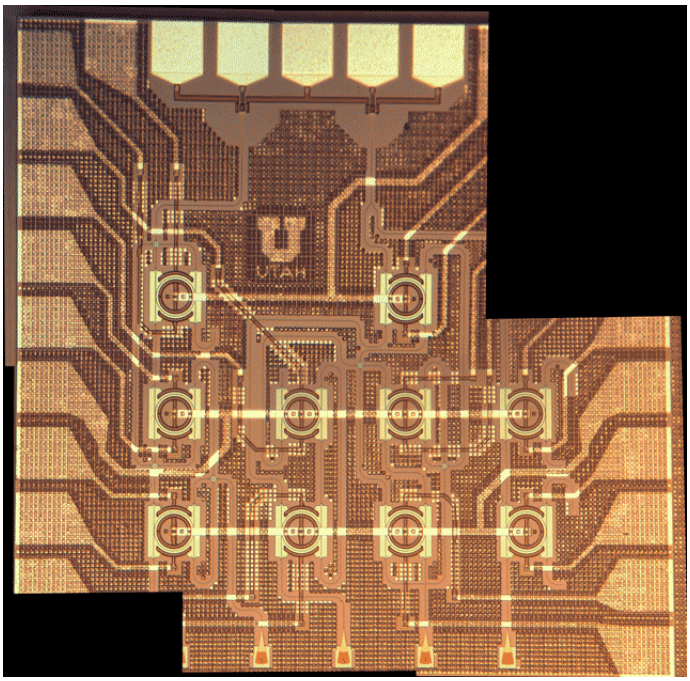


Fig. 1.3 Our silicon photonic 1-bit full adder fabricated through OpSIS

CHAPTER 2

INTEGRATED OPTICS PRELIMINARIES

Integrated optics are used in high speed communication systems for routing signals in the form of light from sources to destinations using optical waveguides [56]. Waveguides are fabricated with layers of different refractive indexes. The waveguide's guiding (film) layer, with refractive index n_f, is sandwiched between substrate and/or cladding layers (n_s and n_c); for $n_s = n_c$ we simply use n_s to describe both layers. The guiding layer has a refractive index greater than the substrate layers, and this enables light to be confined within the structure.

Consider the waveguide depicted in Fig. 2.1. Light is confined within the guiding layer only if the angle of incidence at the $n_f \rightarrow n_s$ boundary exceeds the critical angle θ_c as defined by Snell's law [56]

$$\theta_c = \arcsin\left(\frac{n_s}{n_f}\sin 90\right) \qquad \text{(Critical angle from Snell's Law)} \tag{2.1}$$

Sandwiching the guiding layer between two substrate layers means that should the angles of incidence remain beyond the critical angle at both interfaces, total internal reflection (TIR) will occur within the guiding layer. *Total internal reflection is the primary mechanism behind the guiding properties of optical waveguides.*

The critical angle defines the minimum bend radius necessary to contain light within the waveguide without losses. As waveguides bend, the angles of incidence are affected, possibly leaking light into the substrate should the radii become too small. Waveguides made with greater differences in index of refraction have much smaller critical angles and therefore can tolerate smaller bend radii. Waveguides constructed of silicon and SiO2, i.e., SOI, are excellent in this respect, with a relatively large refractive index difference of 2.0.

2.1 Propagation of Light in Waveguides

Though the ray diagram in Fig. 2.1 is sufficient to describe TIR, it does not account for the electromagnetic wave properties of waveguides. Consider the symmetric slab waveguide depicted in Fig. 2.2. As the electromagnetic waves of light propagate down this structure, the waves reflect

at the boundary of the guiding layer, reflecting at the same incident angle θ due to TIR. For a given angle, the motion of waves down the waveguide defines a k-vector triangle with orthogonal components: 1) the transverse component k_x and 2) the longitudinal component, the *propagation constant*, denoted as β.

The propagating waves interact with each other, causing constructive and destructive interference. If we solve for the system of equations [38]—including materials, waveguide dimensions, and wavelength—we find that only for *discrete* values of β does *constructive* interference occur. These values are denoted the *guiding modes of a waveguide*. For all other angles, the waves destructively interfere with each other and dissipate over distance.

The discrete modes of a waveguide are numbered from the fundamental mode, mode 0. These modes are orthogonal. Depending on the its profile properties, a waveguide may support multiple modes simultaneously. Figure 2.3b depicts the shape of a mode in the vertical and horizontal directions within a 2D waveguide profile; combined, we see a 2D mode profile such as depicted in Fig. 2.3c. The majority of the electromagnetic field is confined within the guiding layer; however, evanescent fields also extend partially into the substrate where they may couple to other structures.

The propagation constant β captures how light propagates through the waveguide materials and structure as compared to the free-space wavelength λ_0. More formally

$$\beta = \frac{2\pi n_{eff}}{\lambda_0} \tag{2.2}$$

where n_{eff} is the *effective index* of the waveguide.

The effective (refractive) index n_{eff} is a function of the mode number, waveguide geometry, material refractive indexes, and the wavelength of light. In effect, values of n_{eff} for the various modes of a waveguide are defining properties of the waveguide. As each n_{eff} is unique for a given mode of a waveguide, it is used synonymously with the given mode for purposes of device design. For simple structures, such as the symmetric slab waveguide depicted in Fig. 2.1, analytical/graphical solutions can be easily found. For more complex geometries, including 2D waveguide profiles, the mode must be calculated using numerical methods provided by tools such as [41], [42].

Electromagnetic waves comprise two fields: an electrical field and a magnetic field. The interaction of these fields is described by a wave function. A plane wave is described using the notation

$$E(z,t) = Ae^{j(kz-\omega t)} \tag{2.3}$$

where A is the wave's amplitude, k is the *propagation constant*, z is distance, ω is the angular frequency, and t is time. Overall, (2.3) describes how the wave moves with respect to space and

varies with respect to time. The waveform of (2.3) is periodic, having a wavelength λ defined over space as

$$\lambda = \frac{2\pi}{k} \tag{2.4}$$

The majority of integrated optical waveguides are planar structure fabricated on or within substrate materials using lithographic and deposition methods. As the dimensions of the component waveguides usually do not vary in height, the width of waveguides and their spatial placement are used for fabrication of devices. A 3D structure, such as the ring resonator in Fig. 2.3d, can therefore be modeled as a 2D structure, simplifying design for many devices.

2.2 Integrated Optic Systems

Figure 2.4 depicts a high-level view of an integrated optics system. We describe the components of this system and their operations; the details of the individual devices can be found in [38]. At the optical inputs of a system are lasers that provide light at the wavelengths the system is designed for, around 1550 nm for SOI systems. For silicon-based processes, this light is usually coupled into the system from outside using fiber couplers or grating couplers. To inject data into the system, modulation devices, such as Mach Zehnder interferometers (MZIs), are used to vary the intensity of the input light. The light is then routed throughout the substrate using waveguides and optical switching devices with electrical switching inputs or in some cases employing all-optical switching.

The routing network also includes passive devices such as waveguide splitters, waveguide crossings, and passive multiplexing devices, such as array waveguide gratings. Splitters divide the input among two outputs with each output receiving half the input power, minus losses. Crossings are necessary for waveguides to cross each other on the single-layer planar substrate with minimal losses; crossings will feature into our physical design work in subsequent chapters. Devices such as array waveguide gratings enable (de)multiplexing of various wavelengths and have been a useful application for 1st-generation silicon photonics.

At the outputs of the system are demultiplexers for multiwavelength systems, photodetectors, and garbage outputs. Waveguides can support ranges of wavelengths, and therefore multiple channels of data may be present on a waveguide that need to be demultiplexed at the output. After demultiplexing, a photodetector (receiver) is required to translate optical signals into electrical signals to read the transmitted data. Such photodetectors utilize materials such as germanium [57], which are incorporated into modern silicon photonics processes [22]. Finally, some routing networks need to dispose of unused light. To prevent interference and noise, the light from these "garbage outputs" must either be routed to the edge of the substrate for disposal or absorbed by a material such as germanium, placed near the exit-point of the waveguide.

2.3 Conclusion

We have described the basics of integrated optics and integrated optic systems. We also note that the integrated optic system in Fig. 2.4 can be modeled as a set of devices interconnected with waveguides. Except at the endpoints, most of the optical devices in the system will be switching or other routing elements. We therefore concentrate our building-block approach on switching (routing) devices such as MZIs and ring resonators. In our behavioral synthesis chapter, these abstract switching building blocks will feature as logical elements. In our physical synthesis chapter we treat these devices as blocks requiring routing. Finally, the waveguide concepts presented in this chapter will feature in the device modeling we present in our thermal-aware resynthesis technique.

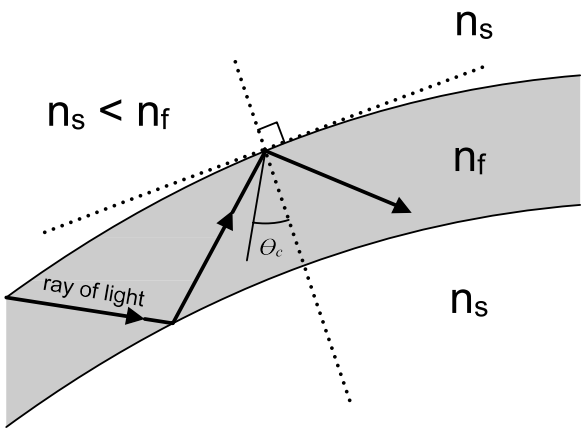


Fig. 2.1 Total internal reflection of light within a waveguide

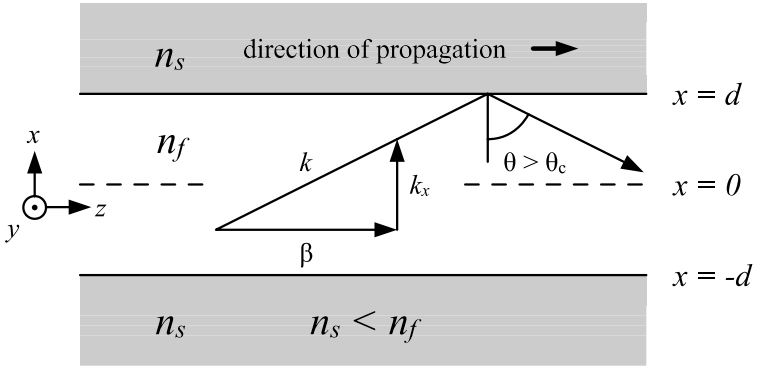


Fig. 2.2 Symmetric slab waveguide

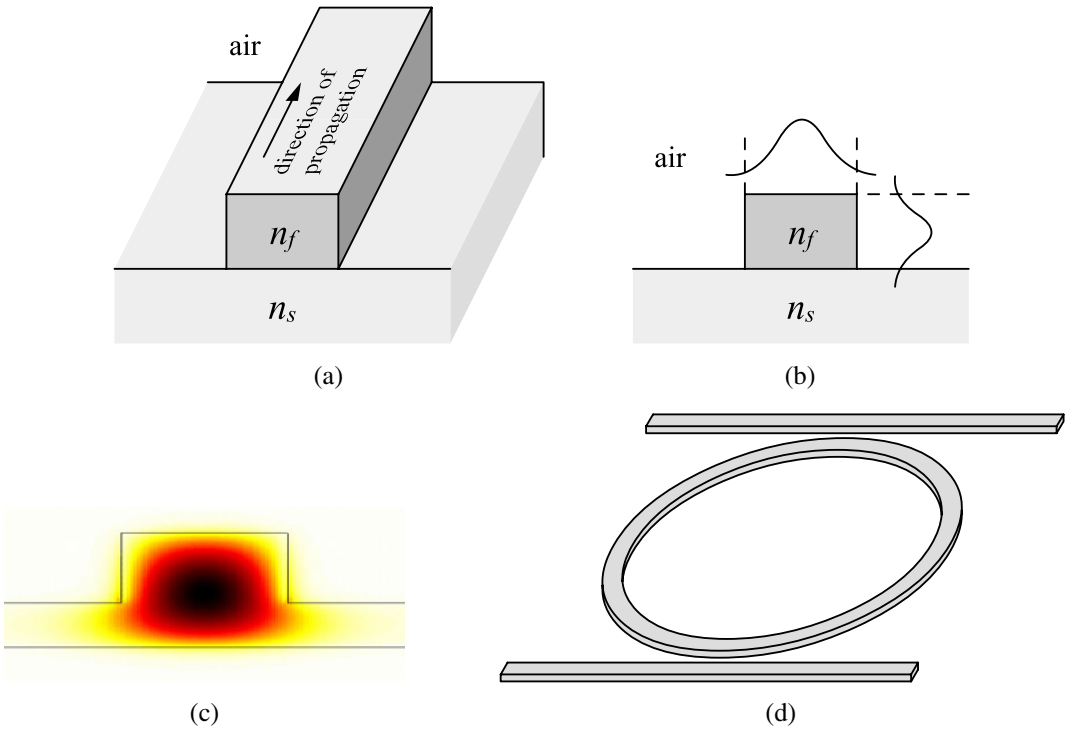


Fig. 2.3 Waveguide and mode shapes: (a) 3D view of a waveguide structure (b) Mode shapes of a waveguide profile (c) Numerically calculated mode shape (d) Ring-resonator modulator

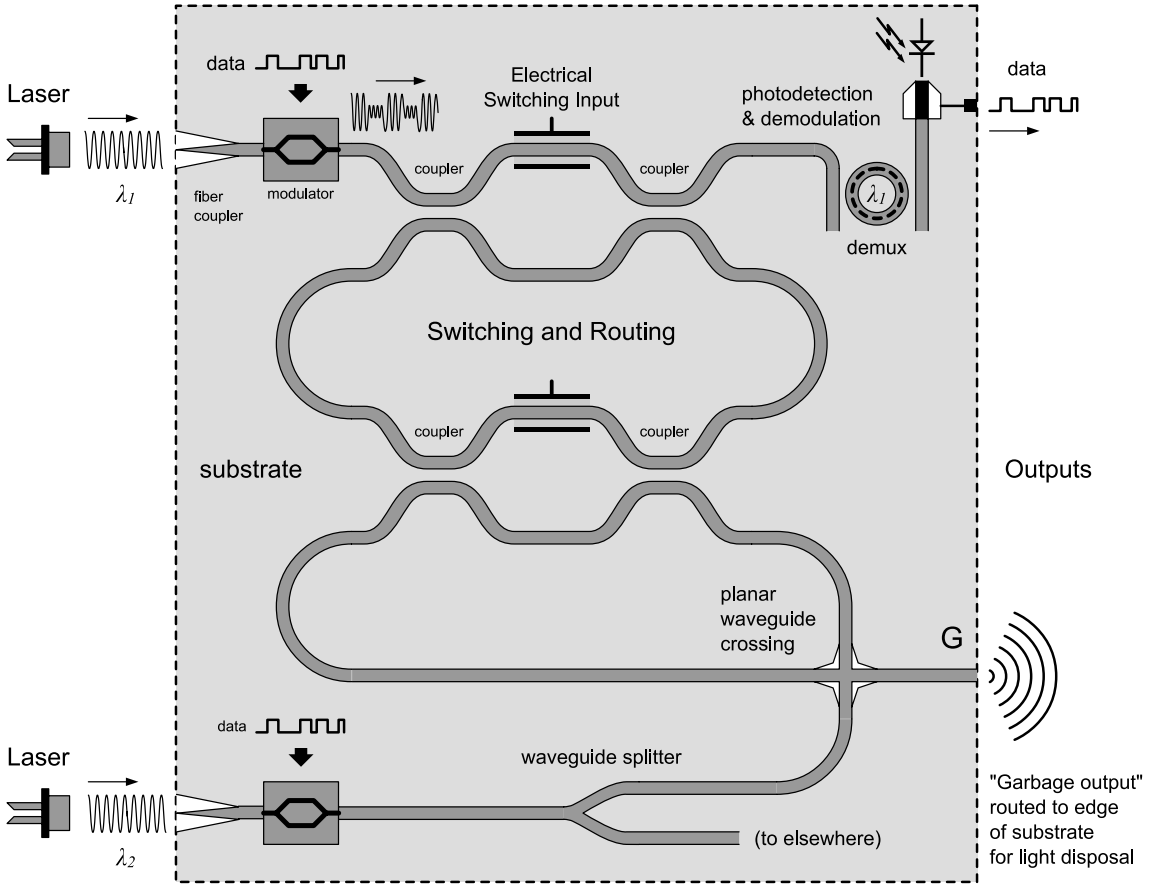


Fig. 2.4 High-level view of an integrated optic system

CHAPTER 3

BEHAVIORAL SYNTHESIS

One of the goals of this work is to develop synthesis techniques that utilize conventional integrated optics devices that can be fabricated with current technology while also being applicable to future design processes. We describe the basic operation of the integrated optic devices we utilize. The constraints of the physical device model are key to the logic synthesis methodology we develop in this chapter.

3.1 Modeling Mach–Zehnder Interferometers

Routing light using waveguides is performed through the use of coupling and controlled interference. Consider the Mach–Zehnder Interferometer (MZI) depicted in Fig. 3.1a created using directional couplers. The paths connected between P and F and Q and G are waveguides. Under certain conditions, when waveguides are brought in close proximity to each other, energy transfers between one waveguide to the other, and vice versa. The couplers in this device are 3dB couplers, dividing and/or combining the signal from both inputs equally between the two outputs. The actual routing is controlled by input S, described by the following equations:

$$\phi_1 = \frac{\omega}{c} \cdot n \cdot L \qquad \phi_2 = \frac{\omega}{c} \cdot (n + \Delta n) \cdot L \tag{3.1}$$

$$\Delta\phi = |\phi_2 - \phi_1| = \pi = \frac{\omega}{c} \cdot \Delta n \cdot L \tag{3.2}$$

where ω is the angular frequency of the light (dependent on wavelength), ϕ_1 and ϕ_2 represents the phase of the light in the two center waveguides, and n is the index of refraction for the waveguide.

Figure 3.1b depicts the MZI in parts, an input S causes a change in refractive index Δn in the region indicated by (1) via heating, carrier injection, or other means. This causes a path-length difference, and therefore a phase difference, between the signals in (1) and (2), causing constructive or destructive interference at the second coupler. A phase difference of 0 or π [56] will route each input *completely* to one output or the other, and the device acts as the controlled crossbar depicted in Fig. 3.2a. Similarly, other designs [16], [58], as depicted in Fig. 3.1c, can be used to reduce the amount of phase-shift needed and the size of the overall device. Changing the refractive

index can be accomplished by using a microheater or more advanced methods such as the Metal Oxide Semiconductor (MOS) capacitors used in Intel's high-speed modulator [19]. Modulation is also possible using devices such as ring resonators. The operation of such devices will be covered in later chapters. In our work, we can utilize either an MZI or ring resonators as an electrically controlled optical crossbar switch to design digital optical logic.

3.2 Our Device Model

The operation of the MZI allows us to model it as a crossbar *gate* that routes light signal completely between two paths depending on the state of **S** and depicts it symbolically in Fig. 3.2a, with its two states Fig. 3.2b and Fig. 3.2c (bar and cross, respectively). The waveguides are sourced by light (logical "1") or darkness ("0"), and the output of a function is read using optical receivers at the end. In our model, the switching input **S** is an *electrical* signal; it is an outside signal that controls the cross/bar configuration and cannot be switched by optical inputs. Connections to **p** and **q**, and **f** and **g** are waveguides, and for simplicity, light is assumed to move from the **p** and **q** side to **f** and **g**. In our model, *an optical signal cannot directly switch a crossbar's **S** input*.[1] More formally

$$\begin{aligned}(S=0) &\Rightarrow (P=F)\wedge(Q=G)\\ (S=1) &\Rightarrow (Q=F)\wedge(P=G)\end{aligned} \tag{3.3}$$

These constraints affect how functions may be composed and imply that the inputs to a crossbar are the primary inputs for that network. Waveguide connections between crossbar gates are depicted symbolically as black "wires." All designs created using the above model can be physically realized, including allowing waveguides to cross each other without interference.

In addition to MZIs, we also utilize *optical splitters,* depicted symbolically in Fig. 3.2d. A splitter divides the light from one waveguide into two output waveguides, each of which contain the original signal, but at half the power (a 3dB loss). In our model, splitters are the only signal degradation mechanism for a given topology, as we assume that there are no losses due to waveguide bends or insertion losses for MZI devices. Such losses are factored into heuristics during the physical synthesis stage, described in the later chapters of this dissertation.

3.3 Previous Work in Integrated Photonic Logic

Early attempts in optical digital logic were designed around the idea of using light in spatially projected computation to take advantage of spatial parallelism [59]. However, the practicality of

[1] Switching a crossbar gate with an optical signal requires an opto-electrical interface comprising an optical receiver unit feeding switching hardware—a system that can be large, expensive, and slow. Designing for such hardware is currently beyond the scope of the synthesis technique applied to this device model.

such designs and devices did not justify their use as they were very slow, large, and expensive; this approach to optical digital logic was generally abandoned.

For optical logic using guided light, investigations into all-optical transistors have also been made and continue to be made [60], [61]; however, these devices are very experimental and often require materials and processes not found in contemporary processes. Ultra-fast switching using nonlinear directional couplers [62], [63], terahertz optical asymmetric demultiplexers [64], nonlinear optical loop mirrors [65], and Sagnac gates [66] have also been explored, but are generally the realm of fiber-based optics. Nonlinear Kerr effects in SOI processes can be exploited [67], but require interaction with special nonlinear cover materials. Investigations into nonlinear, soliton-based logic gates are also promising [9]; however, these devices require either fibers or relatively long (e.g., 5 mm) SOI waveguides [68] to implement.

Optical logic design using crossbar routing devices has been investigated in literature. Shamir, Caulfield, and others investigated the use of optical crossbar gates as *Fredkin gates* [69], [70]. The Fredkin gate model assumes that an optical input can *also* drive the switching input of a gate, allowing the gate to be used in a reversible logic role, but precluding its applicability to our device model. There has also been research in non-Fredkin crossbar gates [4], [6], [7], demonstrating the potential for implementing digital logic using MZIs; however, these are generally confined to small demonstrative circuits that do not scale to larger design implementations and arbitrary logic functions. More recently, techniques such as [71] investigate the integration and routing of optical interconnects; however, such work is for routing, not implementing logic.

We therefore explore methodologies for composing logic functions using MZI crossbar gates that can also scale to larger designs if necessary. We explore how these may be used directly and how their limitations motivate a technique for logic sharing without violating the opto-electrical barrier.

3.4 First Attempts and Crossbar Logic Forms

Static-CMOS benefits from two important properties: metals and semiconductors conduct when physically connected, and logic is restorative in nature. These two properties grant static-CMOS a great level of flexibility for implementing and optimizing logic functions, especially as it allows fanout for multilevel logic implementation. Unfortunately, this flexibility does not extend to optical circuits.

Consider the two networks in Fig. 3.3 implementing functions $f_1 = a + b$ and $f_2 = c \cdot (a + b)$. The first network implements f_2 by using the output of f_1 to drive the switching input of a gate. This is an unworkable design under our model because an optical signal f_1 cannot switch the electrical input of another gate. A more optimal solution is found in the second design Fig. 3.3b, which

uses f_1 as an *optical* input to another gate. This design benefits from using fewer gates, but more importantly, the subfunction is kept entirely in the optical domain. In such a way subfunctions *can* be shared, but with limitations.

- **Waveguide splitters:** The device that enables signal sharing using waveguides is the *waveguide splitter*. A waveguide splitter shares the signal of the input waveguide between two output waveguides, dividing the input power between two outputs, generally with a 50:50 ratio (3dB loss). As the outputs of the splitter have only half the power of the original signal, there are limitations on how many may be used, which can serve as a cost-metric in the design of an optical logic network. Furthermore, as an optical signal, the subfunction may still only be switched and routed further using primary inputs to the network.
- **Garbage outputs:** A "garbage output" is a waveguide output that is not connected to a receiver (a function output), i.e., it is left unused. These unconnected outputs cause problems because the signals, and the light/energy it carries, may interfere with the operation of the network if not properly "disposed." This is demonstrated in Fig. 3.4, which is the visual output of a Finite Difference Time Domain (FDTD) simulation [72] of an MZI device. The FDTD simulation technique models wave propagation through a (discrete) wave medium; Fig. 3.4 depicts the MZI device routing light from the top-left input to the lower-right output. The lower-right output of the device is left unconnected. Light arriving at this unconnected output can do a number of things, including dispersing into the substrate as noise and heat (as shown in the figure as ripples in the substrate) and/or reflecting back into the device, interfering with other signals.

Garbage outputs are problematic and must be properly routed to the edges of the substrate where they can be dispersed away from the logic devices. The additional waveguides needed for this can cause congestion and complicate the overall physical routing of a network. *Every crossbar gate output that is left unconnected is a garbage output.* For example, the network shown in Fig. 3.3b would require three garbage outputs to be routed to the edges of the substrate, leading to a far-less compact design. Minimizing gate count, in general, reduces the number of garbage outputs and is an important part of any synthesis procedure.

With these constraints in mind, we now explore two basic design styles/methods for creating optical crossbar logic networks: BDD-based design and Virtual Gate design. We show how these design styles operate and highlight their abilities as well as limitations. These limitations motivate more advanced approaches using Boolean decomposition as a means to derive designs that may be more optimal and beyond the ability of the other approaches to optimize for. All these described

methods lend themselves to automation and provide a comparison of these approaches near the end of the chapter, using metrics which are described in the coming sections.

3.5 BDD-Based Design

The 2x2 crossbar can be modeled as two multiplexers with complemented inputs. As multiplexers, each crossbar gate effectively implements Shanon's expansion in one variable:

$$f = \bar{x} f_{\bar{x}} + x f_x \tag{3.4}$$

$$output_f = \bar{s} p + s q \tag{3.5}$$

$$output_g = s p + \bar{s} q$$

We can therefore utilize logic structures that employ Shanon's expansion, namely (Reduced Order) Binary Decision Diagrams (BDDs) [44] for direct implementation using crossbar gates.

Consider the Reduced Ordered Binary Decision Diagram (ROBDD) in Fig. 3.5a, which implements two functions: $f_1 = ab + c$ and $f_2 = \bar{a}b + c$, using variable order $a \prec b \prec c$. A dashed line indicates the negative cofactor, and a solid line the positive cofactor, which are connected to the p and q ports of a gate respectively. This is reflected in Fig. 3.5b. A crossbar network can therefore be technology-mapped from the BDD. The BDD's variable-switched function form directly maps to crossbar gate networks and does not violate our crossbar model. In addition, the properties of the resulting network are also directly related to the properties of the BDD structure, including the effects of variable ordering on the canonical structure of an ROBDD.

3.5.1 Salient Features

A BDD-based crossbar network will, in general, have a number of garbage outputs equal to the number of nodes present in the BDD. The physical aspects of crossbar gates also mean that networks cannot take advantage of ROBDD extensions such as complemented edges as the signal in a waveguide cannot be "inverted" without extra hardware; complemented functions will need to be derived as separate BDD function. Common subexpression extraction is possible in the form of shared functions is possible through the use of splitters; however, the effects of the signal degradation must be accounted for.

BDD-crossbar networks are relatively path-delay balanced as they have a feed-forward design topology. The longest path is computed as

$$l_{max} = h \cdot l_0 \tag{3.6}$$

where h is the height of the BDD graph.

Where BDD-based design suffers is in the number of garbage outputs produced by the approach. Each gate has the potential to produce a garbage output that must be accounted for through routing or a light absorbing structure. The canonical structure of ROBDDs can also lead to networks of extremely large gate counts for a given function. Though BDD-based design is attractive for its predictable signal delay, the number of garbage outputs and unpredictability of logic composition in terms of gate counts leads us to abandon this logic composition method for crossbar gate logic. We therefore investigate a composition methodology using "virtual gates."

3.6 Virtual Gate-Based Design

Consider the device networks depicted in Fig. 3.6. We denote these logic composition functions "Virtual Gates." A *virtual gate* (VG) is—functionally and conceptually—a crossbar gate that is switched by a *function,* not necessarily a primary input. The gate is "virtual" in the sense that it is a black box for a function composed of "real" gates—those driven by primary inputs—as well as other virtual gates. A novel form of nesting can be used to compose VG function implementations, where Boolean operators are implemented by replacing child gates with other gates a real or virtual.

A given VG implementation comprises two input waveguide ports **p** and **q** connected by waveguides and crossbar gates to two output ports **f** and **g**. The nesting operation comprises the Boolean operator forms depicted in Fig. 3.6 and is illustrated in Fig. 3.7a where two AND virtual gates are nested within an OR virtual gate, creating the final function $ab + cd$. Evaluation of a VG, given a primary input assignment, involves assigning **p** and **q** inputs logical 0 and 1, respectively, and applying *cross* or *bar* configurations to gates as defined in Fig. 3.2. The output of the function is detected at **f**, with $\mathbf{g} = \neg\mathbf{f}$.

The process of composition is illustrated in Fig. 3.7a, where a function $f = ab + cd$ is implemented by replacing (or *nesting*) the gates of an OR function with VGs implementing $a \cdot b$ and $c \cdot d$. The result is depicted in Fig. 3.7b.

While it may seem strange to see feedback loops in device designs, the physical devices can indeed implement self-feedback. As an experiment, the model for the AND gate depicted in Fig. 3.6a was simulated in a 2D FDTD simulator OptiFDTD®by Optiwave Software; the visual output[2] of which can be seen for $a = 1, b = 0$ in Fig. 3.8. The signal from the top-left crosses in the top gate, but passes through in the bottom gate, returning to the top gate where it crosses again to appear in the top-right output.

---

[2]Note that there are differences from the virtual gate diagram: the bottom two ports are swapped because the waveguides are not crossed in the center and the "light" source is positioned at the p input rather than at the q input.

3.6.1 Salient Features

Networks composed of virtual gates have exactly two optical inputs, p and q, and two outputs, f and g, as the entire network is, in itself, a virtual gate; in addition, for a given function, a maximum of one garbage output is created. The existence of a complete logic enables virtual gates to implement any logic function using crossbar gates *comprising only primary inputs.* This includes factored functions and any other single-output representation using Boolean operators. Control signals (S) are connected via the primary inputs of the function. The f port implements the function, and $g = \neg f$. Furthermore, the total number of *real* gates is the number of primary literals in the original logic expression the network is derived from.

Virtual gates also suffer from very unbalanced signal paths, depending on the state of the switches, with the potential for a signal to traverse every waveguide present in a VG network. The maximum signal path l_{max} is roughly computed as

$$l_{max} = 2 \cdot p \cdot l_0 \tag{3.7}$$

where p is the number of operators in the virtual gate, and l_0 is a "unit length" of waveguide. This is based on the fact that all virtual gate operators connect two gates (virtual or real) by two waveguides, and a signal could possibly traverse all paths to reach the destination. For example, the network in Fig. 3.7a would have a $2 \cdot 3 \cdot l_0 = 6l_0$ long maximum signal path, which is close to the longest possible signal path from p to f with variable assignment $\{a,b,c,d\} = \{1,0,1,0\}$ at $5l_0$. The value l_{max} is a reasonable rough estimate; it can be further refined by estimating routing distances for operators and physical network topology.

3.6.2 Expression Sharing

The major limitation of designing with virtual gates is that the nesting of gates *prevents the extraction/sharing of arbitrary common subexpressions (CSE)*. For example, in Fig. 3.9 one cannot simply share the ab term from $f = ab + bc$ for use with another gate; assignments such as $abcd = \{1,1,1,1\}$ will cause all crossbar gates to assume a cross-configuration, isolating the top input of the h-gate from the optical inputs of the network. In effect, any operator employing feedback for its inputs can produce an undefined state. Only the XOR operator does not exhibit this behavior as it has no feedback, but XOR-based CSE is not well studied.

Despite the versatility of the virtual gates to implement any logic function, the inability of a virtual gate network to share subexpressions easily implies that the overall network can be very large due to replicated expressions. Furthermore, a virtual gate network has the potential for long signal paths—traversing every waveguide in the network.

3.7 XOR-Based Common Subexpression Extraction

Virtual gates are theoretically capable of implementing any single-output Boolean function in a factored form. This, however, has some caveats: the inability to drive gate-inputs using optical signals implies that only splitters may be used to share expressions. This places severe constraints on VG expression sharing, namely that sharing is not possible using operators connecting operands via *loops* (e.g., AND and OR operators). Common subexpressions connected via AND or OR operators must be *reimplemented* (replicated) wherever they appear as discussed in the previous section.

Expression sharing using VGs is only possible through the use of the exclusive-OR (XOR) operator as it does not contain loops in its construction. This has its limitations, however: expression sharing is *hierarchical*. Consider the three VGs XORed together in Fig. 3.10a. If we attempt to share the output of the b VG, the output is not the *function* b, but rather the function formed by b and its inputs: $a \oplus b$. Sharing the function b directly would require a separate b gate driven only by 0 and 1 at its inputs. Therefore, common subexpression extraction, as implemented in CMOS technologies, cannot be applied to VG networks. However, it is still possible to perform expression sharing by means of XOR decomposition, the structure of which is depicted in Fig. 3.10b.

3.7.1 XOR-Based Expression Sharing

Our goal in expression sharing is to reduce the overall literal count—and therefore gate count—by sharing functionality. The network topology in Fig. 3.10b depicts f_0 and f_1 sharing a common subexpression P through the relationship in (3.8). Ideally, this relationship will reduce the total literal cost of the design.

$$f_0 = P \oplus Q_0 \qquad f_1 = P \oplus Q_1 \tag{3.8}$$

$$f_0 = (P \oplus m) \oplus (Q_0 \oplus m) \qquad f_1 = (P \oplus m) \oplus (Q_1 \oplus m) \tag{3.9}$$

Reducing literal count utilizing the XOR operator is not as straightforward as with AND or OR operators [73], [74] and more difficult considering the feed-forward topology in Fig. 3.10b. We approach this problem as one of adding or removing cubes from the subfunctions P, Q_0, and Q_1.

Consider the case where an arbitrary term m is XORed with the right-hand-sides of both equations of (3.8). In order to balance the equations, we use the XOR identity $a \oplus a = 0$, requiring that m must be added again to each of the equations. If we group the terms as depicted in (3.9), what can be taken from the result is that simultaneously adding a term m to all three functions P, Q_0, and Q_1 does not change the functionality of f_0 and f_1. We can choose terms such that one or more of the functions are simplified as a means of *optimizing* the overall expression-sharing VG network, as we will see in the following example.

3.7.2 Motivating Example

Consider the binary-coded-decimal (BCD) to 7-segment display in Fig. 3.11a, which converts a BCD to a visual representation of a number by turning segments on or off (1 or 0) depending on the value (0–9) of the BCD ($x_3x_2x_1x_0$). The truth table for segments 0 and 1 (S_0 and S_1) is depicted on the right side of Fig. 3.11a (unlisted rows are assumed to be zero). The table is mapped to Karnaugh maps (K-maps) depicted in Fig. 3.11b, allowing us to derive the prime implicants for the functions and the resulting sum-of-products (SOP) equations below the K-maps. Through this method, the total literal count for the two SOP expressions is 21 literals, requiring 21 crossbar gates if implemented as virtual gate networks.

We now decompose S_0 and S_1 into functions P, Q_0, and Q_1 as depicted in Fig. 3.10b, where $S_0 = P \oplus Q_0$ and $S_1 = P \oplus Q_1$, and initially assign $P := 0$, $Q_0 := S_0$, and $Q_1 := S_1$. At this point, the network is essentially the same as implementing S_0 and S_1 separately. Now consider the case where we XOR an expression k with P, Q_0, and Q_1, where k is the intersection of minterms contained in Q_0 and Q_1. This operation has the effect of cancelling those minterms in P, Q_0, and Q_1 that are also contained in k and adding them if not. This new set of functions is actually *less* optimal than the original (depicted in Fig. 3.11c), because some of the larger cubes are broken up in the XOR operation. These less-optimal functions can, however, be improved by repeating the operation using minterms 1 and 6, affecting all three functions P, Q_0, and Q_1, resulting in the K-maps and functions in Fig. 3.11d. The final set of functions uses only 10 literals total—*11 gates less* than implementing the original functions separately. This example demonstrates the potential for good common expression sharing by "adding" and "removing" *minterms* from the decomposition functions. The same operation can also be extended to cubes in general.

Note that while the decomposition is XOR-based, the subfunctions P, Q_0, and Q_1 are implemented as VGs in any form—sum-of-product, factored forms, etc. The most optimal form is chosen for VG implementation. This is a novel feature of our decomposition technique as compared to conventional XOR-based optimization methods.

3.7.3 Limitations of Contemporary XOR-Based Synthesis Techniques

XOR-based optimization is well studied in literature. Techniques such as [73], [74] are designed to minimize Exclusive-Sums-of-Products (ESOPs) expressions with exact and fast heuristic methods. ESOP expressions are, however, very limiting compared to VG networks as they comprise only AND-XOR terms. Decomposition methods have also been explored using graph structures and concepts such as x-dominators [75] to find structural XOR relationships. However, an XOR decomposition can only be extracted if found on the graph structure; an XOR decomposition with expression sharing always exists in our approach. [76] addresses some shortcomings of

previous decomposition methods by finding linear relationships between subfunctions of form $f = g_1h_1 \oplus g_2h_2$, thereby reducing the area of XOR-based logic functions. While this performs an XOR decomposition, it does not create common subexpression sharing "by design." The technique described in [77] applies a heuristic method of sharing subfunctions of positive-polarity Reed–Muller expansions for Toffoli gate synthesis. However, as with other Toffoli synthesis methods [78], expression sharing of this type is incompatible with our approach because expression outputs cannot be shared across the opto-electro barrier. We therefore present a technique for finding XOR decompositions for VG networks while simultaneously performing common subexpression sharing.

3.8 Multi-Output Expression Sharing

Boolean functions are represented using ROBDDs [44]; this enables a more compact representation while allowing efficient XOR-based manipulations. Rather than using minterms to manipulate functions, as in the motivating example, we use cubes derived from the BDDs. The number of literals in the function (our metric) is the sum of all literals from the cubes of a BDD. It should be noted that while the literal count of the BDDs is used as a metric during synthesis, the BDD is used solely as a function-manipulation data-structure, not as a technology-mapping/implementation structure.

Two functions can be decomposed into a structure depicted in Fig. 3.10b. This can be extended to a multilevel decomposition by repeating the process hierarchically. The function in Algorithm 1 implements this procedure as a top-level function decomposition from a multioutput design, returning a decomposition tree of subfunctions representing the optimized design.

The algorithm selects most "compatible" functions f_0 and f_1 using BESTPAIR(), where compatibility of functions is ranked such that the number of shared variables is maximized and the number of function-exclusive variables is minimized—increasing the probability of producing a useful decomposition. Using f_0 and f_1, the algorithm attempts to find a P, Q_0, and Q_1 decomposition that can replace f_0 and f_1 as a branch in the tree. When a decomposition improves the literal count, the result is mapped into the decomposition tree, the stems of the decomposition (Q_0 and Q_1) are removed from the function pool (F_0), and the root P is added to F_0 for further decomposition. The result of this procedure, applied to all segments of the BCD-to-7-segment display, can be seen in Fig. 3.12. Segment outputs S_0, S_1, S_2, S_3 and S_6 are able to benefit from multilevel sharing, where outputs S_4 and S_5 are only able to share functionality with each other and are implemented separately.

The actual XOR decomposition is performed by Algorithm 2, taking two functions f_0 and f_1 as inputs and producing an improved decomposition, or FALSE if no decomposition could be found.

$\theta()$ counts the number of literals in a given set of BDDs. Variable N_0 is a chosen maximum number of passes; we use $N_0 = 130$.

The XOR decomposition technique works by applying cubes to the decomposed functions such that net literal count is reduced. SEED() returns all cubes from $Q_0 \cup Q_1 \cup Q_0 + Q_1 \cup Q_0 \oplus Q_1$ to provide an initial pool of cubes to optimize with. In each iteration, a cube is selected from C and XORed with P, Q_0, and Q_1 to attempt to reduce the net literal count. Cubes from the resulting decomposition are then added to the cube pool C, further increasing the available cubes that can be used. These cubes are repeatedly used until no improvement is found. The technique then tests the result against the best found result, storing it if there is improvement. A new starting point is then chosen to repeat the process. This continues for a chosen number of passes N_0.

An important part of our approach is the ability to hill-climb out of local minima. This comes in two forms: the first occurs at lines 16 and 18 and is important for allowing the algorithm to apply cubes to the decomposition even when they cause no literal-count improvement. To prevent deadlocks, this is allowed only for E_0 number of times ($E_0 = 10$ in our implementation). This gives the technique more flexibility in finding a better decomposition. The second method allows a restart at a point based on the best decomposition and a cube that caused the largest effect (line 22) on the decomposition. The process then repeats and continues until there are no more passes left.

After a complete decomposition is performed for a design, the subfunctions of the decomposition tree are implemented as optimized factored-forms and mapped to VGs. The final decomposed multi-output design is implemented as a tree of XOR-decomposed functions, in the same type of structure as seen in Fig. 3.12. We evaluate this technique's efficacy on a number of logic designs in the next section.

3.9 Experimental Results

The crossbar logic synthesis technique described in Section 3.8 is applied to a number of logic designs from the ACM/SIGDA (i.e., MCNC) logic synthesis benchmark suites [79]. We also include the BCD-to-7-segment design. Two designs (cm138a and cm42a) saw no change via our technique and are not included in the table.

The results of the technique's application is seen in Table 3.1. The original literal count L_{orig} represents the number of literals counted for implementing all outputs separately as VGs. The decomposed literal count L_{decomp} represents the number of literals of the decomposed network after applying our technique. Also included is the number of subfunctions (**#funcs**) implemented as VGs in the decomposed implementations.

Overall, most designs enjoy reduced literal counts when the decomposition is applied (negative ΔL), in some cases orders of magnitude differences. An increase or no change in literal counts for

some designs can be attributed to discrepancies between the literal counts of the BDD-functions used in the technique's internal metrics and the actual implementations of those functions as VGs.

3.9.1 Limitations

Our synthesis procedure does not allow for electro-optical interfaces (receivers and transmitters) except at the start and endpoints of the circuit. However, in larger systems, these functional blocks comprise parts of the overall design that must be interconnected. A more extensive synthesis procedure is needed to partition larger circuits at electro-optical transceiver boundaries, with individual blocks implemented via synthesis techniques such as those presented in this chapter.

3.10 Application to a Fabricated Design

Our virtual gate methodology is the basis of the optical 1-bit full adder design submitted for fabrication with the OpSIS [22] foundry and depicted in Fig. 1.3. The gate network for the *sum* and C_{out} functions are depicted in Fig. 3.13. Though the XOR-based logic synthesis technique was applied to the pair of functions, the resulting gate count was increased due to the additional gates required to synthesize the *sum* function as compared to its compact 3-gate XOR representation.

As a means to save area, the fabricated design utilizes ring resonator-based [16], [58], rather than MZI directional coupler-based, devices as switching elements. These devices are logically equivalent; however, the Q and G ports of single-ring resonators are swapped as depicted in Fig. 3.14.

The resulting ring-based full adder network is depicted in Fig. 3.15. Though the ring resonators reduce area, routing complexity is increased due to the $Q-G$ port swap. The waveguides must also cross each other in order to connect to their destination ports. How such routing complexity is accounted for is the subject of subsequent chapters.

3.11 Conclusion and Future Work

This chapter describes design and synthesis methods for implementing digital logic using integrated optical devices that function as crossbar switches. We have shown a design methodology for constructing arbitrary logic functions using VGs and present an XOR-based methodology for expression sharing for multi-output designs. The efficacy of our synthesis techniques is shown on a number of logic designs, often with large improvements.

3.11.0.1 Future work

This synthesis procedure is limited to implementations that do not incorporate electro-optical transceivers. As part of a more extensive synthesis procedure, partitioning techniques have to be

explored to enable larger designs to be implemented as a series of interconnected subfunctions designed using techniques such as presented in this chapter. However, the physical design of the optical network is an integral part of such partitioning. Parameters such as signal degradation from splitters, routing congestion, and delay balance will ultimately decide how circuits are partitioned for separate implementation. Therefore, it is required to explore ways to integrate this technique with automated layout and routing using the same "building block" concept used for synthesis.

This will enabling physical design to be coupled with automated synthesis and allow for further refinement of the synthesis process. In addition, such a physical framework will enable us to explore how other techniques can be incorporated into our logic design. It may be interesting, for example, to investigate how different wavelengths operate within the same logic structures and whether this enables greater resource sharing with techniques such as wavelength division multiplexing. Our end goal is to produce a framework for design automation for integrated optics. The framework will perform both logic synthesis for integrated optics as well as lead to future directions in physical design and layout.

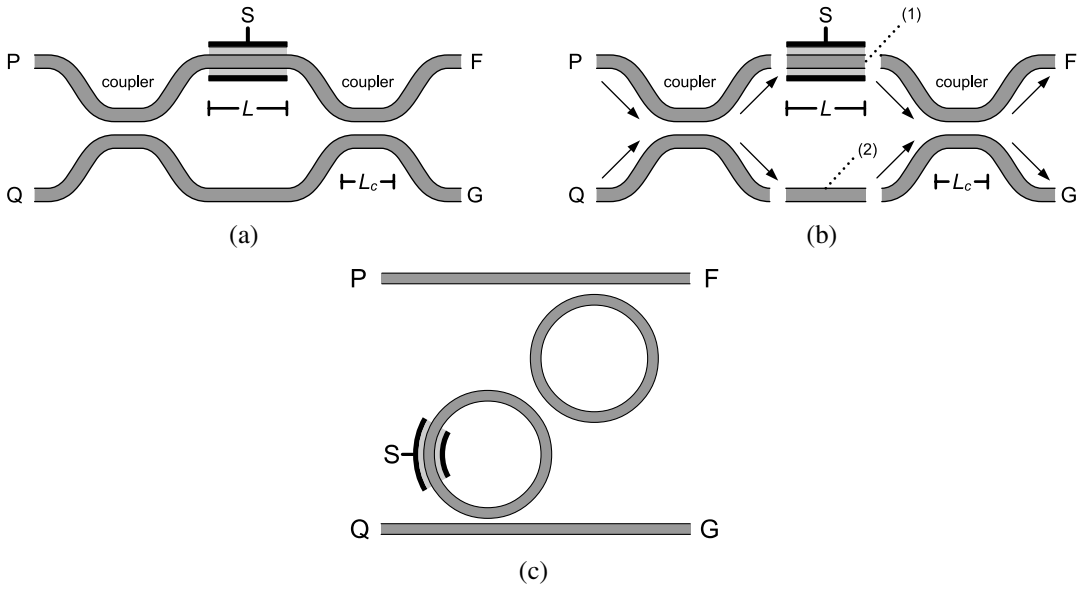


Fig. 3.1 Mach–Zehnder Interferometer (MZI) routing devices:
(a) Directional coupler-based (b) Directional coupler MZI in parts
(c) Ring-resonator-based

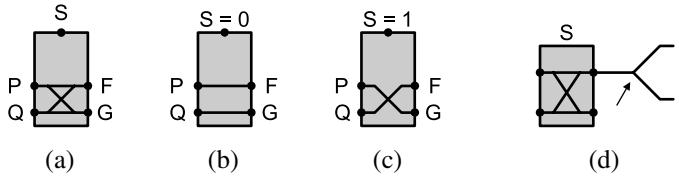


Fig. 3.2 Crossbar switches: (a) Gate (b) Bar (c) Cross (d) Splitter

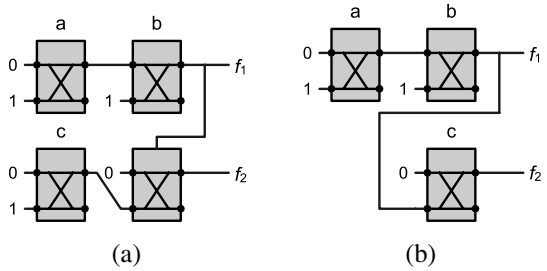


Fig. 3.3 Two configurations for $f_1 = a + b$ and $f_2 = c \cdot (a + b)$:
(a) Incompatible design (b) Compatible design

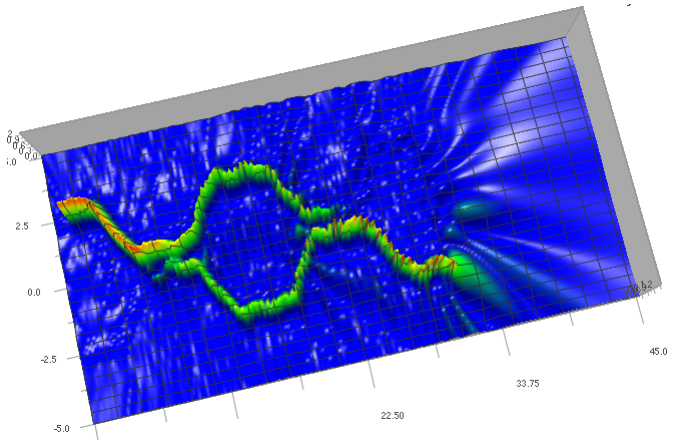


Fig. 3.4 Dispersion of light into the substrate from a garbage output

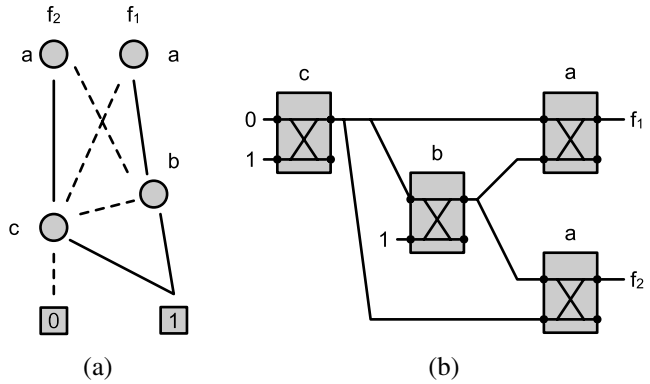


Fig. 3.5 BDD-based design for $f_1 = ab + c$, $f_2 = \bar{a}b + c$: (a) BDD graph (b) Resulting BDD-based design

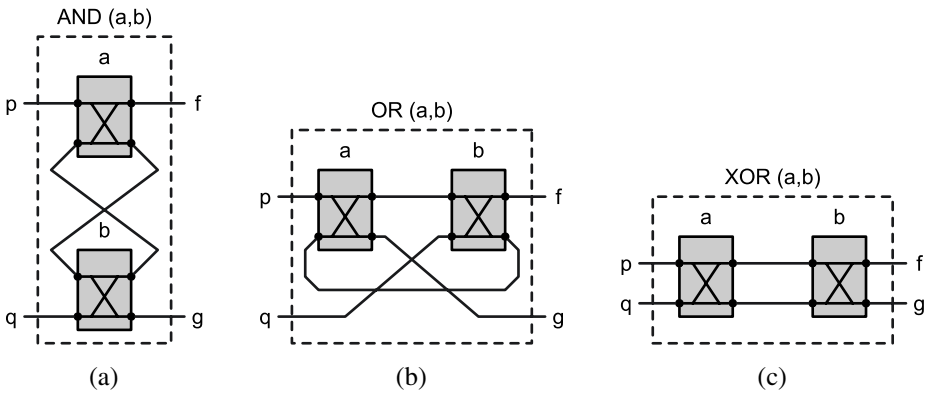


Fig. 3.6 Virtual gate functions for 2-input Boolean operators: (a) AND (b) OR (c) XOR

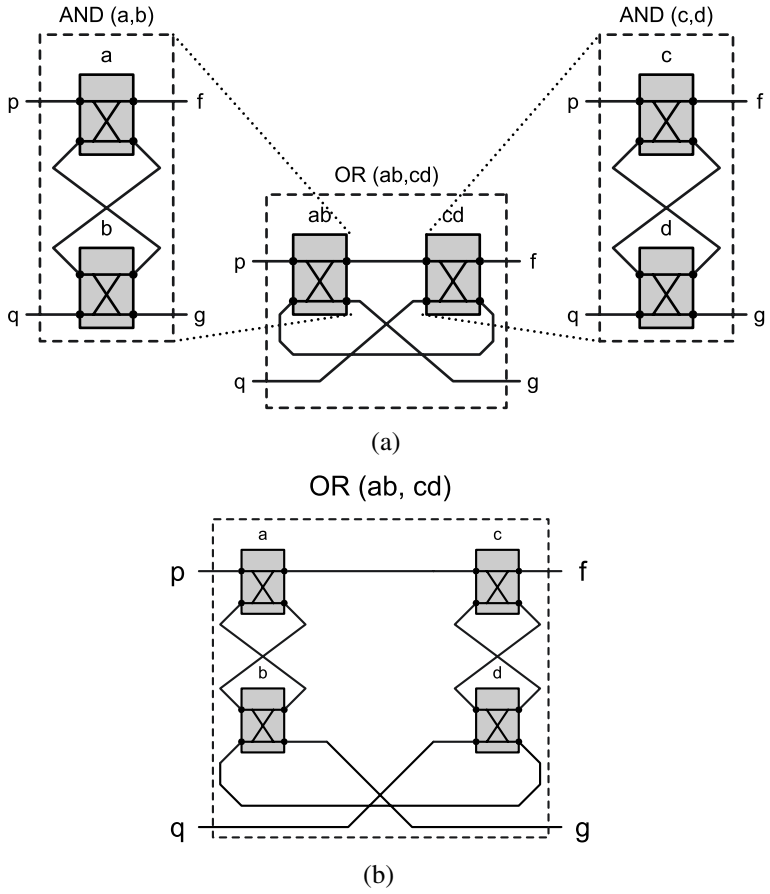


Fig. 3.7 Composing functions with virtual gates: (a) Virtual Gates implementing $f = ab + cd$ (b) Resulting network

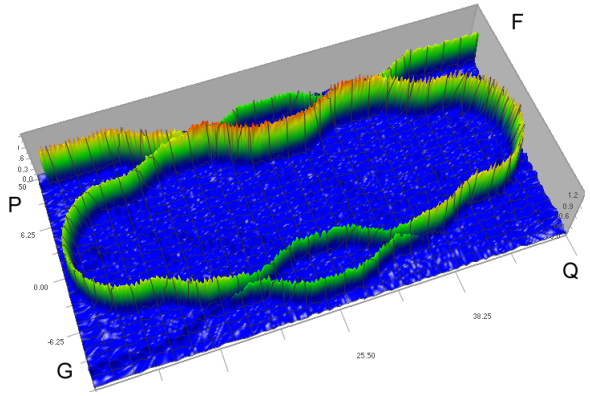


Fig. 3.8 FDTD simulation of an AND virtual gate

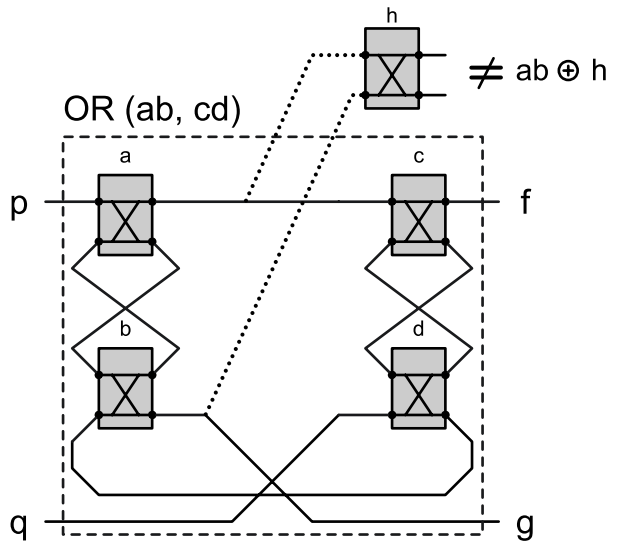


Fig. 3.9 Internal functions of virtual gates cannot be shared

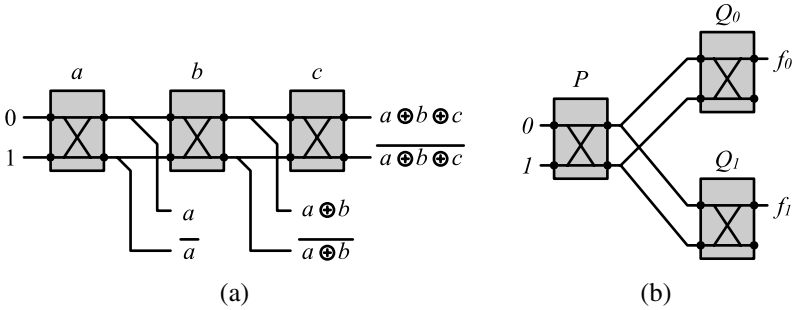
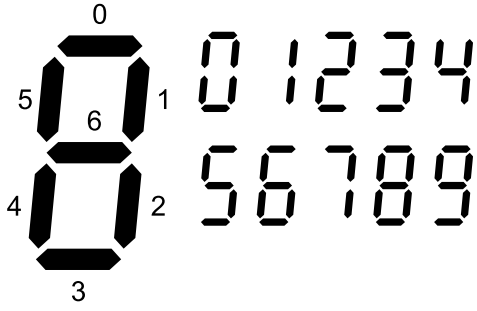


Fig. 3.10 Virtual Gate expression sharing: (a) XOR expression sharing is hierarchical (b) XOR decomposition structure

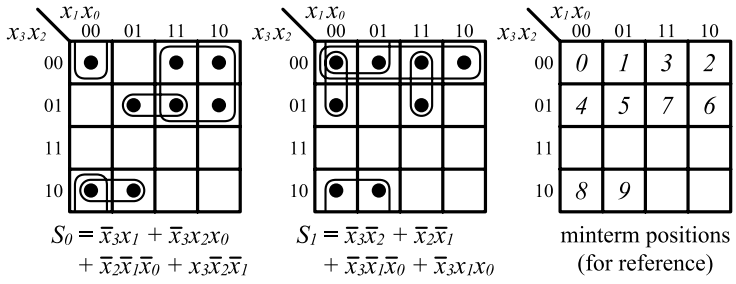
Algorithm 1 Function Optimization

```
function OPTIMIZEDESIGN(D:design)
    MAPTOTREE(D → T);
    F0 := FUNCTIONSFROM(D); Up := ∅;                    ▷ Up = Used pairs
    while (F := (x ∈ F0, y ∈ F0) : x ≠ y, (x,y) ∉ Up)) ≠ ∅ do
        (f0, f1) := BESTPAIR(F); Up := Up ∪ (f0, f1);
        if (B := XORDECOMP(f0, f1)) ≠ FALSE then
            MAPTOTREE(B[P], B[Q0], B[Q1] → T);
            REMOVE(f0, f1)FROM(F0);
            F0 := F0 ∪ B[P];
        end if
    end while
    return T;
end function
```

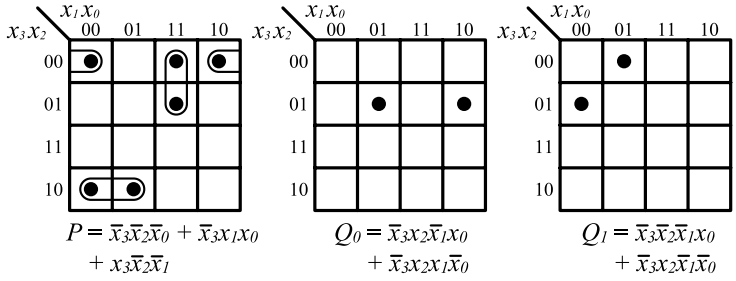


	$x_3x_2x_1x_0$	S_0	S_1	S_2	S_3	S_4	S_5	S_6
0	0 0 0 0	1	1	1	1	1	1	0
1	0 0 0 1	0	1	1	0	0	0	0
2	0 0 1 0	1	1	0	1	1	0	1
3	0 0 1 1	1	1	1	1	0	0	1
4	0 1 0 0	0	1	1	0	0	1	1
5	0 1 0 1	1	0	1	1	0	1	1
6	0 1 1 0	1	0	1	1	1	1	1
7	0 1 1 1	1	1	1	0	0	0	0
8	1 0 0 0	1	1	1	1	1	1	1
9	1 0 0 1	1	1	1	1	0	1	1

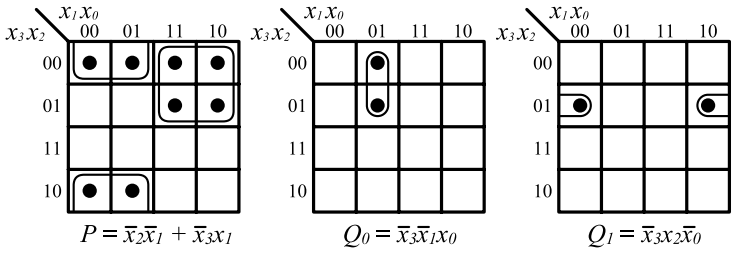
(a)



(b)



(c)



(d)

Fig. 3.11 XOR operations on BCD-to-7 segment display function:
(a) 7-segment display and truth table for segments 0 and 1
(b) Karnaugh maps for S_0 and S_1 (unmarked grids = offset) (c) The original P, Q_0, and Q_1 after XORing with k (d) Optimized P, Q_0, and Q_1

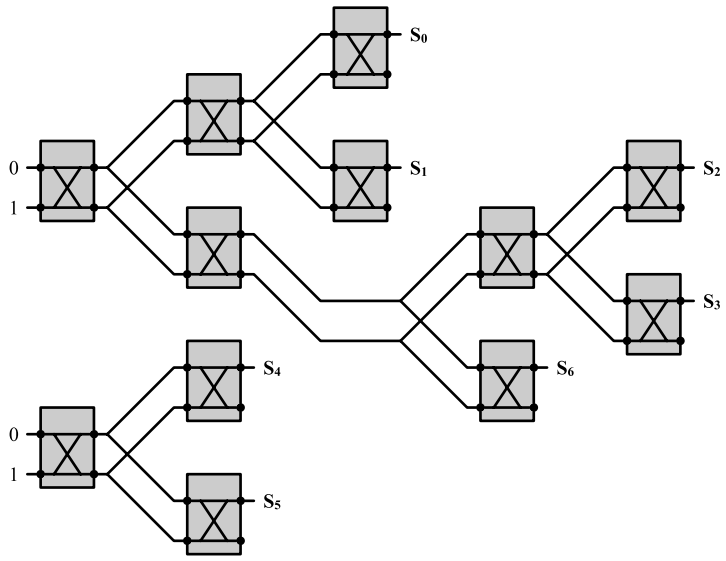


Fig. 3.12 BCD-to-7-Segment complete decomposition

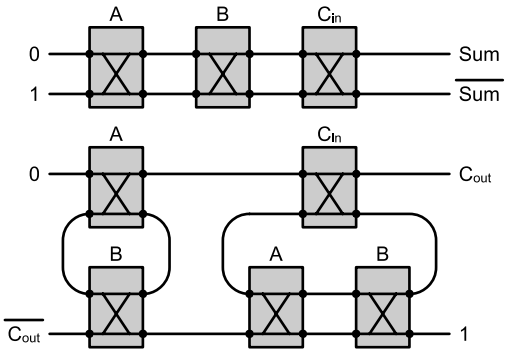


Fig. 3.13 1-bit full adder constructed from virtual gates

Fig. 3.14 Ports of two functionally equivalent optical logic building blocks: (a) Directional coupler-based logic gate (b) Ring-based logic gate

Algorithm 2 XOR Decomposition

function XORDECOMP(f_0, f_1)
  $P := 0$; $Q_0 := f_0$; $Q_1 := f_1$; $L_0 := \theta(\{P, Q_0, Q_1\})$;
  $best := [P, Q_0, Q_1, L_0]$; ▷ Current best results
  $N := N_0$; ▷ N = Passes left; N_0 = total passes
  $V[] := \emptyset$; ▷ $V[e]$ maps $e \rightarrow SetofCubes$;
  $U_v := \emptyset$ ▷ U_v = used V cubes
  while $(N > 0)$ **do**
    $C := \text{SEED}(\{P, Q_0, Q_1\})$; ▷ C = cubes;
    $U_c := \emptyset$; ▷ U_c = used cubes
    $L_1 := \theta(\{P, Q_0, Q_1\})$; ▷ Starting # literals for pass
    $L := L_1$; $E := E_0$;
    repeat
      $m := \text{REMOVECUBEFROM}(C)$; $U_c := U_c \cup m$;
      $p := P \oplus m$; $q_0 := Q_0 \oplus m$; $q_1 := Q_1 \oplus m$;
      $v := \theta(\{p, q_0, q_1\}) - \theta(\{P, Q_0, Q_1\})$;
      if $(v < 0)$ **or** $((v = 0)$ **and** $(E > 0))$ **then**
        $P := p$; $Q_0 := q_0$; $Q_1 := q_1$; ▷ Accept the change
        $E :=$ **if** $(v = 0)$ **then** $E - 1$ **else** E_0;
        $L := L + v$;
      end if
      $C := C \cup (\text{CUBESOF}(\{p, q_0, q_1\}) \setminus U_c)$;
      $e := |\theta(p) - \theta(P)| + |\theta(q_0) - \theta(Q_0)| + |\theta(q_1) - \theta(Q_1)|$;
      if $m \notin \text{CUBESOF}(V)$ **then**
        $V[e] := V[e] \cup m$; ▷ Map cube's effect e to cube
      end if
      if $(C = \emptyset)$ **and** $(L \neq L_1)$ **then**
        $C := U_c$; $U_c := \emptyset$; $L_1 := L$; ▷ Retry until no change
      end if
    until $(C = \emptyset)$;
    if $L_1 < best[L_0]$ **then**
      $best := [P, Q_0, Q_1, L_0]$;
      $N := N + 1$; ▷ Reward improvement with extra passes
    else
      $N := N - 1$;
    end if
    $m_0 := (c \in V[e] : \text{largest}(e), c \notin U_v)$;
    $P := best[P] \oplus m_0$;
    $Q_0 := best[Q_0] \oplus m_0$;
    $Q_1 := best[Q_1] \oplus m_0$;
  end while
  if $best[L_0] < L_0$ **then**
    return $best$;
  else
    return FALSE
  end if
end function

Table 3.1 Logic synthesis benchmark results

Design	In	Out	L_{orig}	L_{decomp}	ΔL	# funcs
5xp1	7	10	294	160	**-134**	15
alu2	10	6	25645	899	**-24746**	9
alu4	14	8	6227	4906	**-1321**	13
apex4	9	18	15967	4154	**-11813**	32
b1	3	4	16	9	**-7**	3
b12	15	9	1847	146	**-1701**	13
bcd7seg	4	7	132	35	**-97**	11
bw	5	28	955	314	**-641**	43
c8	28	18	200	406	+206	27
cc	21	20	147	136	**-11**	27
clip	9	5	888	736	**-152**	9
cm162a	14	5	85	125	+40	9
cm163a	16	5	43	65	+22	8
cmb	16	4	76	48	**-28**	4
cps	24	109	7156	5332	**-1824**	152
cu	14	11	91	71	**-20**	11
decod	5	16	80	65	**-15**	16
duke2	22	29	2174	2220	+46	43
ex1010	10	10	86694	5433	**-81261**	19
ex5p	8	63	60960	902	**-60058**	79
f51m	8	8	317	109	**-208**	11
i1	25	16	82	88	+6	17
inc	7	9	744	176	**-568**	14
lal	26	19	184	196	+12	25
ldd	9	19	427	141	**-286**	25
misex1	8	7	122	93	**-29**	12
misex2	25	18	188	175	**-13**	24
misex3	14	14	17971	13232	**-4739**	25
misex3c	14	14	5006	6892	+1886	27
pcle	19	9	87	131	+44	14
pcler8	27	17	199	420	+221	22
pdc	16	40	208008	41269	**-166739**	79
pm1	16	13	67	72	+5	16
rd53	5	3	144	74	**-70**	5
rd73	7	3	840	249	**-591**	5
rd84	8	4	3288	465	**-2823**	6
sao2	10	4	532	250	**-282**	7
sct	19	15	141	265	+124	22
spla	16	46	141815	3372	**-138443**	69
sqrt8	8	4	155	120	**-35**	6
sqrt8ml	8	4	1382	44	**-1338**	7
squar5	5	8	387	70	**-317**	11
table3	14	14	7021	4446	**-2575**	25
tcon	17	16	48	48	0	24
ttt2	24	21	337	544	+207	31
x2	10	7	87	132	+45	9
z4ml	7	4	62	114	+52	6

L = # literals

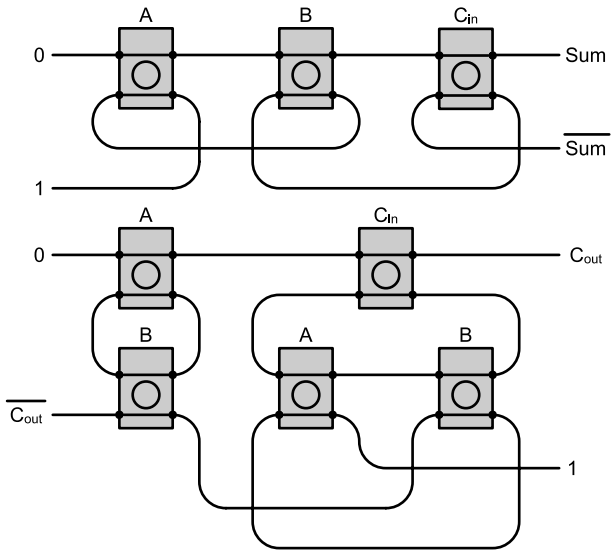


Fig. 3.15 Ring-resonator-based 1-bit Full Adder

CHAPTER 4

PHYSICAL SYNTHESIS METHODOLOGY FOR INTEGRATED OPTICS

As the availability and applications of integrated optics expand, the need for automated design space exploration, optimization, and physical synthesis of integrated electro-optical systems is also beginning to appear. For this reason, the Electronic Design Automation (EDA) community is investigating how automatic design space exploration techniques can be adapted to the photonics domain [2], [71], [80]–[83].

We also take a step forward in this direction and investigate *physical design automation for integrated electro-optical circuits and systems*. We show that an EDA-style design methodology—i.e., placement, global routing, detailed routing—is applicable to optical layout and routing, and techniques/algorithms used for VLSI physical design can also be suitably adapted. In this chapter, we highlight how constraint models, design rules, and optimization criteria will drive and govern physical design automation techniques for hybrid electro-optical system integration.

4.1 Previous Work

Design automation is a relatively new concept in integrated optics. For this reason, not much work has been done. At the physical level, [81] demonstrates a *full-custom layout* of photonic structures using a commercial CMOS-based layout editor (Cadence Design Systems Virtuoso). Waveguide curves are discretized at a fine level into rectangular geometry, enabling waveguides to be represented in a format that traditional foundries accept. This methodology is significant in that it provides a building-block pathway for producing foundry-ready layouts and masks for non-Manhattan device geometries (rings, arcs, waveguide curvature). However, "design automation" is essentially absent, and the design must be conceived of and optimized manually. Similarly, the commercially available photonics CAD suite [41], [42] provides a framework for physical device design, analysis and simulation engines for performance analysis of optical design components. However, automated techniques for design space exploration during physical synthesis—*automated*

floorplanning, placement, waveguide routing while optimizing for physical parameters such as insertion-loss, bend-loss, phase coherence issues, etc.—are not available.

Recently, [71], [82] present techniques for global optical interconnect synthesis. Such techniques analyze the routing problem at different levels of abstraction than the techniques presented in this work. Once global routing is performed, local routing is necessary to complete routing. At this routing level, a channel router may be utilized to ensure crossing-minimality, as well as heuristically minimal bends. The work of [83] presents high-level, run-time calibration and reconfiguration techniques to reduce power usage in on-chip interconnect networks. The technique employs multiple redundant optical devices to which subchannels can be remapped in cases where it would reduce overall tuning power. Automatic place-and-route for wavelength-routed optical NoCs is presented in [84]. This technique helps designers bridge the gap between network topology schemes and their physical implementations.

4.1.1 Motivation

The main motivation for investigating this problem stems from physical design of integrated *electro-optical logic circuits* [2], [4], [7], [85]. Consider the Graphic Data System (GDS) layout of our OpSIS-fabricated [22] full adder depicted in Fig. 4.1. Eight (8) ring resonators are arranged into columns by a device placer such as to minimize area as well as routing complexity. The column arrangement induces the presence of vertical routing regions between device columns denoted as **channels**, with device connection-points, denoted as **ports**, facing the channels. Interchannel waveguides are used to allow routes between devices in other channels.

Such circuits are complex in their device interconnections, often featuring high device counts and large amounts of feedback loops. These designs comprise a set of predesigned optical devices—modulators, switches, splitters—placed on a planar substrate, connected together via waveguides. For example, in our previous work [2], our multilevel logic synthesis methodology for implementing logic demonstrates how optical designs can scale beyond the ability of custom design. The physical synthesis of such applications now has to be addressed.

The chapter is organized as follows: 1) we first describe the design constraints of our problem formulation; 2) we then outline our EDA-style design flow methodology; 3) the details of each step of the design flow are described, including how such methods are adapted for optics; and 4) how design rules for waveguide routing are accounted for in the routing grid.

4.2 Design Constraints

At the physical automation level, we identify signal power and substrate area as our core guiding metrics.

4.2.1 Signal Power

Signal power is the primary guiding metric in our methodology. All devices, including bulk waveguides, have insertion losses, measured in decibels (dB). Our assumption is that these losses are precharacterized through device-analysis (FDTD, etc.) for the following type devices:

- **Predesigned devices [device-specific]** (e.g., modulator devices, switches, splitters, etc.). Losses are characterized from inputs to outputs. For example, waveguide splitters have their signal power from the input effectively halved at each output (a 3 dB loss).
- **Waveguide crossings [0.1–0.2 dB / crossing]** Per-crossing losses are on the order of 0.1–0.2 dB per crossing [86]–[88], affecting both crossing waveguides.
- **Waveguide bends [0.001–0.3 dB / bend]** Losses are dependent on the inherent waveguide properties (materials, geometry, etc.), radius of curvature of the bend, and surface roughness due to fabrication [89]–[91].
- **Bulk waveguides [0.01–2 dB / cm]** As these losses are extremely low (dB per *centimeter*, e.g. 0.03 dB/cm [92]), we consider bulk waveguides essentially *lossless*.

Losses due to the presence of predesigned devices are effectively fixed. Therefore, the design automation problem concerns itself with designing within the permitted losses *between* such devices—the routing fabric. We identify three main routing loss mechanisms in descending importance: 1) *waveguide crossings,* which induce a relatively large fixed loss per crossing; 2) waveguide bends, especially bends close to the minimum radius of curvature; and 3) bulk waveguides, which generally have low losses; however, surface roughness can induce losses over larger distances for smaller waveguides.

4.2.2 SOI Waveguides

Si-photonic waveguides, with their large refractive index differentials, provide strong mode confinement, and therefore bends can be much sharper, saving area. While waveguide bends can be effectively lossless given a large enough radius of curvature, accepting small per-bend losses can be advantageous in reducing the area occupied by a bend [90]. The choice of minimum routing grid size can therefore affect the weighting of metrics used to guide the routing, whether losses due to bends, waveguide crossings [86], or area.

4.2.3 Area

Many optical devices, such as those used for switching, are designed such that their input and output ports appear on only opposing sides. This feed-forward device design often extends to the device networks as a whole, resulting in overall networks that are very wide. Wide substrates may not be desirable when integrating optics into designs, and a more suitable aspect ratio may need to

be enforced. The side-effect of this is that devices must be rearranged on the substrate in a manner that can affect interdevice locality as well as increase waveguide routing complexity. This becomes an important part of the placement phase of our methodology.

4.3 Methodology

We propose the following methodology for the overall physical design problem for integrated optics. As depicted in Fig. 4.2c, predesigned optical devices are represented as rectangular blocks (a) that are arranged (placed) in fixed-width columns (b). Such a placement gives rise to *vertical routing channels* (c), which are routing regions that separate the placed devices. Waveguides are routed between devices at "ports" (d) that face the channels. For ports in different columns, these waveguides may pass through *horizontal routing channels*, as depicted in (e). While the substrate is planar, waveguides may also cross each other perpendicularly (f) without sharing signals.

Overall, the physical design methodology requires that the problem be solved in three steps:

- Placement of optical switching devices into columns, i.e., a grid-based layout.
- Global routing of waveguides that connect these devices. Global routing solution will determine the overall routing topology of all the nets.
- Detailed routing of all the nets, which manifests itself as a well-defined channel routing problem.

While this methodology is analogous to that employed in the VLSI domain, the design and optimization constraints imposed by the optical technology are different. Any CAD solution to this problem will have to incorporate such technology specific constraint models and design rules.

4.4 Device Placement

Predesigned optical devices are placed into columns. Consider the layout of devices in Fig. 4.2a. While devices maintain ports on only their left and right sides, connections may be made to any other device in the network by routing through vertical columns and between columns. In such a manner, connectivity is preserved, but the overall network has a smaller aspect ratio. This transformation does not come without issues: the column arrangement may affect the locality of connected devices, which in turn affects routing congestion and losses due to routing length, crossings, and bends. The problem of device locality and connectivity is not limited to optics and has been studied extensively for VLSI chip planning. Placement techniques, such as those used for row placement and chip floorplanning [93], can therefore be employed for placing devices within an optical substrate.

The placement of devices into columns enables us to utilize routing techniques designed for such placement strategies. In our applications we use the Capo placer [93] to arrange devices in rows given a specific aspect ratio. Connected devices are localized as much as possible, reducing

congestion. The Capo placer is used directly, without any optics-specific constraint optimization. While optics-specific placement techniques are desirable, especially thermal-aware placement, it is beyond the scope of this dissertation. In contrast, routing solutions for waveguides is one of the important contributions of this work.

4.5 Waveguide Routing

Routing is performed in two phases: 1) *global routing,* which determines the general routing path and horizontal routing channel placement and 2) *detail routing,* which is formulated as *planar waveguide channel routing*.

4.5.1 Global Routing

Global routing determines the high level topology a signal may take through the channels from source to destination. The chosen routes induce bends and crossings with other nets. The optimization goal of the global router is to minimize losses due to waveguide crossings and waveguide bends. In addition, global routing also takes into account overall net lengths and routing congestion. Global routing is explained in detail in Chapter 5.

4.5.2 Channel Routing

The global router provides a set of vertical routing channels with net/port connectivity, such as depicted in Fig. 4.2c. At this stage, detail routing is performed, determining the actual placement of horizontal and vertical connections within the vertical channel. Consider the routing channels depicted in Fig. 4.2b. The channel routing area is a grid between the pins on either side of the channel, where waveguides are routed between pairs of pins. Traditional VLSI channel routing seeks to minimize the area of a fully routed channel. In our channel routing techniques, we optimize for crossings and bends, with channel height a subsequent metric. The details of our channel routing approaches are found in Chapter 6.

4.6 Design Rules: Routing Grid Realization

The result of routing algorithms must be transformed into the physical waveguide layout. This entails converting the routing grids into waveguide bends satisfying the material bend constraints, which are generally defined in terms of minimum radius of curvature and coupling distance.

4.6.1 Mapping Routing Grids to Waveguides

A rectilinear routing grid is realized as waveguides by converting all 90° grid transitions to 90° waveguide bends. This requires that such bends complete within a quarter of the routing grid. This

is illustrated in Fig. 4.3b where a horseshoe-shaped bend utilizes two 90° waveguide bends, each taking place within a quadrant of the routing grid. This mapping represents the smallest grid that can be suitably used for complete routing grid flexibility.

The physical routing can also exploit the spacing between curves at the corners of grids. These "knock-knee" style bends, as depicted in Fig. 4.3c, enable additional track sharing—potentially reducing the overall number of tracks needed for a routing. For example, in the solution depicted in Fig. 4.4b, the knock-knee bends between signals C-E, F-G, D-I, and G-J allow each respective pair to occupy the same track, with the net effect of reducing the total number of tracks to four (4). Routing techniques enabling knock-knee track sharing must account for shared rectilinear grid locations, e.g., Fig. 4.4a, during channel construction.

The waveguide's minimum radius of curvature r has an important role in determining the routing grid's minimum size. In some cases, r may be chosen for area reduction, at the expense of per-bend losses [90]. For example, to enable knock-knee routing patterns, the distance w_c in Fig. 4.3c must be sufficient to prevent significant coupling between waveguides.

4.7 Conclusion

We have presented physical design automation methodology for silicon nanophotonic circuits and systems. Automation will be key to large scale system synthesis. We demonstrated that traditional VLSI physical design flows of placement, global routing, and detailed routing can be adapted to the optics domain. While this methodology is analogous to that in the VLSI domain, the design and optimization constraints imposed by the optical technology are different. We have described the design constraints and optimization criteria that are imposed by such optical technology and show how they can be incorporated into the placement and routing formalisms.

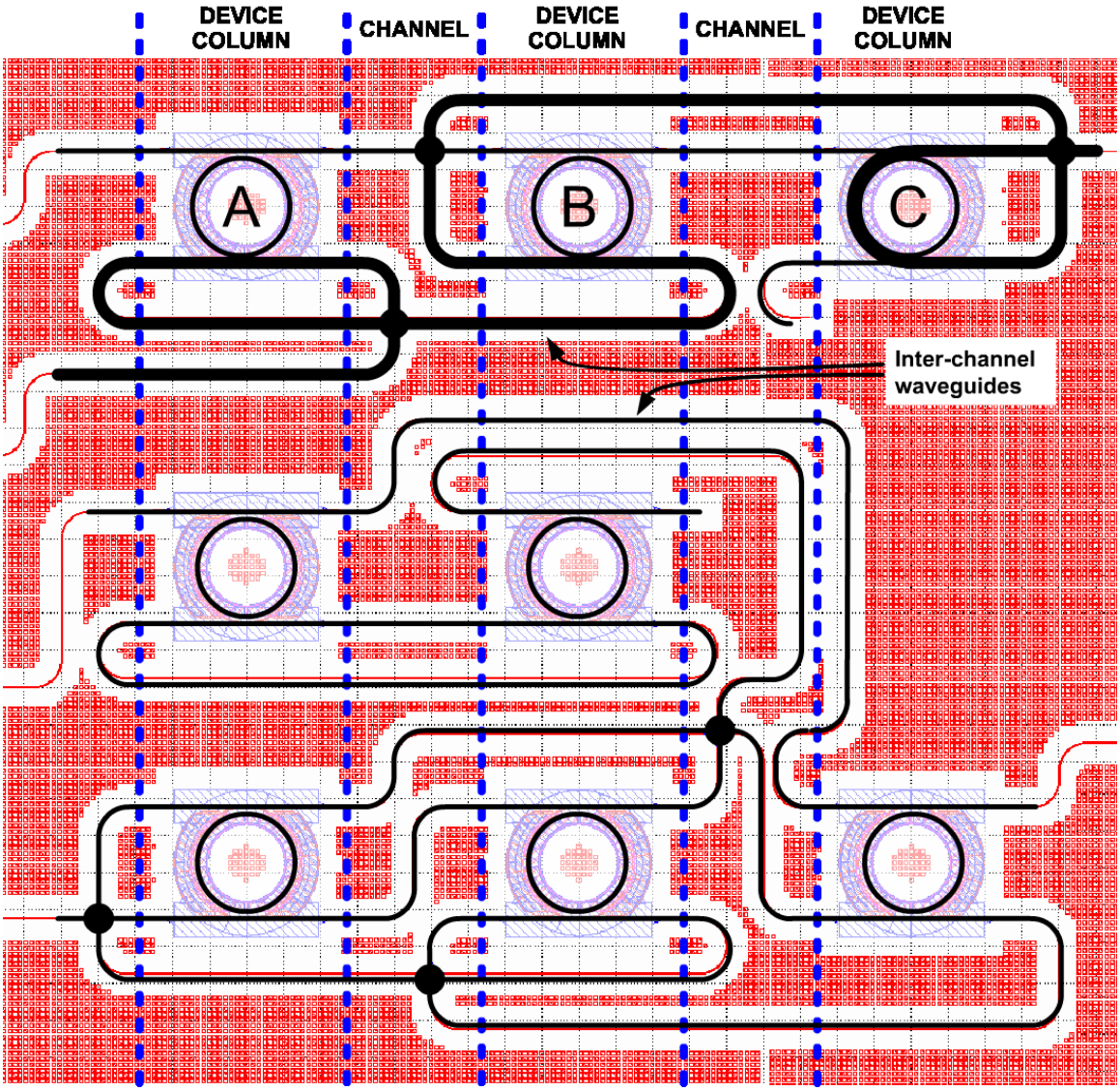


Fig. 4.1 Routing channels of optical GDS layout

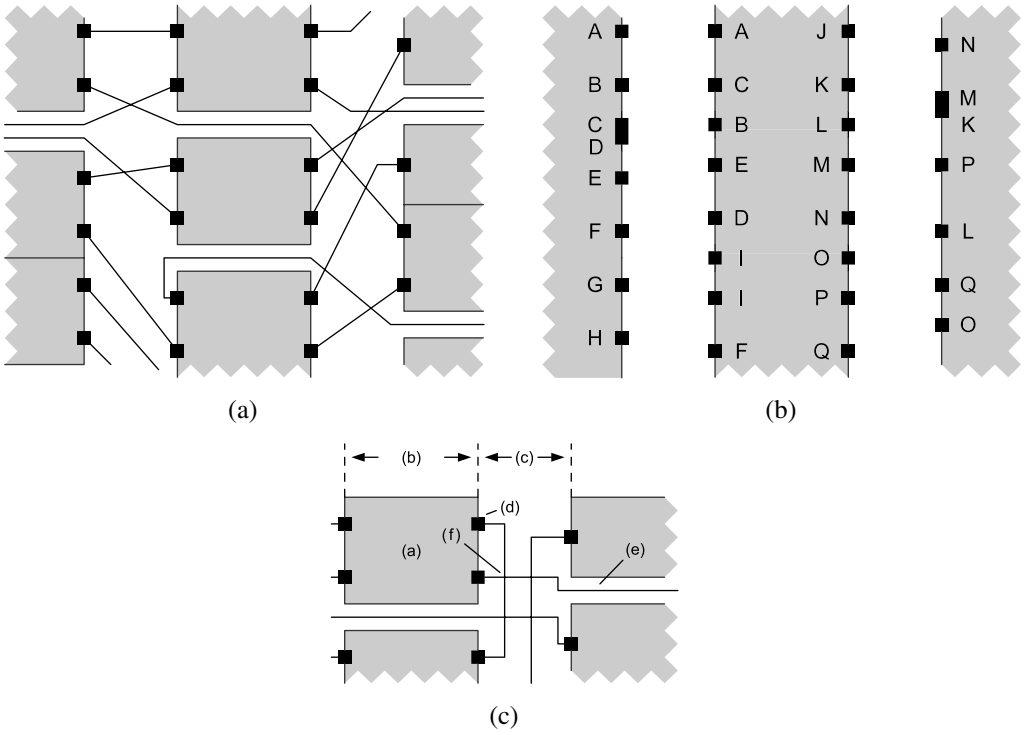


Fig. 4.2 Physical Design Methodology: (a) Columns of optical devices, and global routes (b) Resulting channels for detailed routing (c) Ports, routes and channels

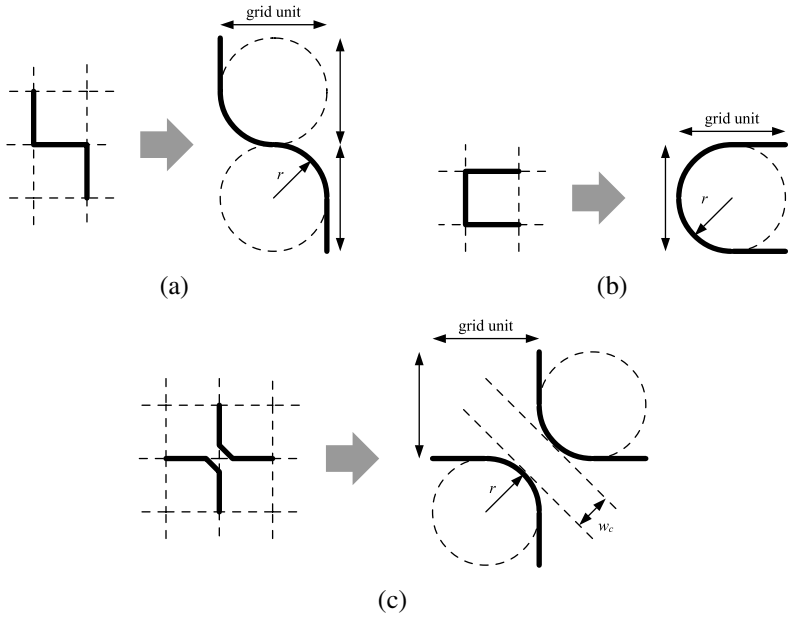


Fig. 4.3 Conversion of grid units to waveguide curves: (a) S-shaped grid to bends (b) Horseshoe-shaped grid to waveguide bends
(c) Knock-knee grid with 90° bends, radius of curvature r, and minimum coupling distance w_c

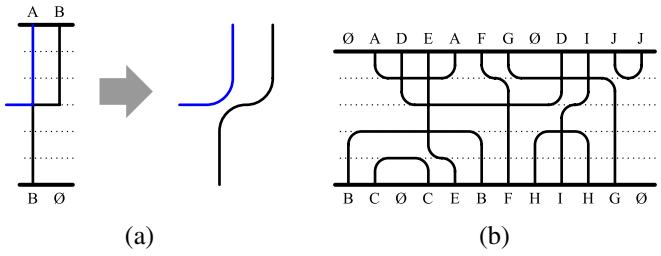


Fig. 4.4 Knock-knee model for for grid spacing: (a) Shared grid corners enable knock-knees (b) Channel routing incorporating knock-knee bends (4 tracks, 8 crossings)

CHAPTER 5

GLOBAL ROUTING FOR INTEGRATED OPTICS

After placement, routing determines the interconnect fabric that connects placed devices together from their respective ports. For large designs of many routes, direct routing of the interconnect fabric is an intractable problem. Therefore, routing is broken into two phases: global routing and detailed routing. Global routing provides the high-level overall placement of routes throughout the device network, while detailed routing determines the localized routing necessary to complete routing.

We present a global routing formulation as a necessary tool for producing crossing-aware, high-level routing solutions for subsequent detailed routing. This global router also incorporates a number of important concepts from VLSI global routing in order to produce realistic solutions.

5.1 Global Routing Problem Formulation

Given a placed set of devices, the global router provides a high-level routing solution that optimizes/satisfies the following:

- **Wire length:** Routes should reduce wire length in order to meet timing constraints, reduce area, and avoid congesting zones.
- **Congestion:** Areas with large numbers of routes are difficult to route in the subsequent detailed routing phase. Congestion also affects such factors as heat generation, which can further compound signal delay and defects. Capacity constraints can limit the number of routes occupying a given area.
- **Overflow:** A routing solution should not produce solutions that require more area than is provided. Should this be an impossible task, device blocks may require different placement.

The result of global routing is a solution that is passed to the detailed routing stage for localized routing.

5.2 Previous and Contemporary Work

Global routers are important for effective routing in the VLSI domain and employ a variety of techniques to complete routing. Sequential-type routing [46], [47], [94], [95] perform routing by sequentially routing nets within areas using iterative negotiation-based rip-up and reroute as well as pattern routing techniques. Such techniques employ heuristics on net ordering and also perform rip-up and reroute techniques to refine the solution. Routers also incorporate concurrent global routing to provide a routing solution for a given set of routes simultaneously. The benefit of this approach is that all routes are accounted for in the process of finding a solution, instead of possible back-tracking in routing using rip-up/reroute-type techniques. The problem is generally solved using mixed integer linear programming (MILP); however, due to the size of most routing problems, techniques such as [46] break down the problem into small hierarchical subsets in order to make concurrent routing feasible. Via-minimization is incorporated into global routers [96], which reduces area and routing overhead in multilayer designs.

Despite the advanced state of VLSI global routers, there are limitations in their applicability to the optical routing domain. VLSI routing is inherently multilayer, and VLSI global routers are designed to take advantage of multiple layers in order to produce routing solutions. As such, global routers are also not designed to minimize crossings. Minimization techniques for metrics such as vias, though applicable to bends, cannot be applied to waveguide crossings as a single via can facilitate multiple crossings due to multiple layers. Also, while 2-pin nets can utilize less complex direct-search techniques such as maze routers and A*-search algorithms [45], the interaction of optical waveguides due to crossings requires that any approach utilize techniques such as concurrent-type routing.

We therefore propose our own MILP-based model for global routing for integrated optics. Our model incorporates constraints and metrics for waveguide crossings, bends, and route length. We also include constraints to account for and control congestion. Our model is then applied to a number of routing problems producing routing solutions.

5.3 Routing Using Mixed Integer Linear Programming

Global routing comprises an MILP formulation with coefficient weights derived from a pre-analysis of the routes corresponding to nets. The coefficient weights encode signal losses induced by chosen routes of a given net and also losses caused by the *combination* of chosen routes—i.e., interroute losses. The overall structure of the MILP formulation is covered first; the details of the pre-analysis are covered subsequently.

5.3.1 MILP Formulation

The global routing problem is formulated on a graph G comprising a set of grid-edges E derived from layout of the device placement. For example, the routing regions between devices in Fig. 5.1 produce a set of edges connecting between port-endpoints; the details of G for a channel-based placement will be explained in a later section.

A set of m unique nets are routed in the graph by selecting one of n_i unique *routes* between the net's corresponding endpoints. This set of routes for net i is R^i. More formally

$$r_k^i = k\text{-th route of } R^i, \text{ where } 1 \leq k \leq n_i \tag{5.1}$$

$$x_k^i = \begin{cases} 1 & \text{if net } i \text{ is routed using route } r_k^i \\ 0 & \text{otherwise} \end{cases} \tag{5.2}$$

$$\sum_{k=1}^{n_i} x_k^i = 1 \tag{5.3}$$

Boolean (binary) variables x_k^i denote whether a given route r_k^i is selected. Only one route $r_k^i \in R^i$ may be chosen for a given net in a given routing, as enforced by (5.3).

5.3.2 Grid-Edge Capacity Constraints

In order to limit congestion, we introduce grid-edge capacity constraints. A route r_k^i comprises a set of edges E_k^i defining the route in the routing-grid G. Grid-edges may be shared by multiple routes; however, only a limited number of route-edges may occupy a given grid-edge e. This *grid-edge capacity* is denoted as e_{max} and is enforced using the following relation:

$$\sum_{i=1}^{m} \sum_{k=1}^{n_i} x_k^i \cdot x_{k,e}^i < e_{max} \tag{5.4}$$

where

$$x_{k,e}^i = \begin{cases} 1 & e \text{ is an edge of } r_k^i \\ 0 & \text{otherwise} \end{cases} \tag{5.5}$$

5.3.3 Route Cost Constraints

A given route r_k^i has a cost α_k^i associated with it, comprising 1) a *static* cost $\alpha_{k,static}^i$, representing the power loss due to physical properties such as bends and the length of the route and 2) *interroute* costs, representing the losses associated with the interaction of the route with other routes—i.e., waveguide crossings. Each waveguide crossing causes signal degradation, affecting both nets involved, and it is possible that two nets will have more than one crossing per assigned route. Therefore, for two different nets i and j, with respective routes r_k^i and r_l^j, we assign a crossing-loss-coefficient $\alpha_{k,l}^{i,j}$ representing the total cost of selecting both routes in terms of total crossings.

This coefficient only applies when both routes are selected, and implemented using a conjunction equation, (5.7).

$$\alpha_k^i = \alpha_{k,static}^i \cdot x_k^i + \sum_{j}^{j\neq i} \sum_{k=1}^{n_i} \sum_{l=1}^{n_j} \alpha_{k,l}^{i,j} \cdot x_{k,l}^{i,j} \tag{5.6}$$

$$x_{k,l}^{i,j} = x_k^i \wedge x_l^j \quad = \begin{cases} 2 \cdot x_{k,l}^{i,j} - x_k^i - x_l^j & \leq 0 \\ x_k^i + x_l^j - x_{k,l}^{i,j} & \leq 1 \end{cases} \tag{5.7}$$

$$A^i = \sum_{k=1}^{n_i} \alpha_k^i \qquad A^i < A_{max}^i \tag{5.8}$$

A given net i has a cost A^i associated with it, equal to the total cost of the net's chosen route—static plus the sum of all interroute costs. This cost is constrained below a maximum tolerated A_{max}^i, which is predetermined by the optical design specifications. As only a single route is chosen for a given net, A^i is simply the sum of the route costs for the given net, (5.8). By construction, any solution found will satisfy the maximum cost constraints for all nets.

5.3.4 Minimization Function

The final minimization function is a sum of all net costs. The function incorporates per-net weights W_i in order to prioritize certain nets over others during optimization. The minimization function is implemented as

$$\textit{minimize:} \sum_{i=1}^{m} W_i \cdot A^i \tag{5.9}$$

5.3.5 Route Pre-Analysis

The MILP formulation described earlier relies on a route pre-analysis that determines the cost-coefficients for all routes and interroutes. The pre-analysis comprises: 1) the derivation of a graph representing the routing channels with weights representing costs due to edge traversal; 2) the derivation of multiple routes per net for analysis and recording their static costs; and 3) the analysis of interroute costs.

5.3.5.1 Graph and route derivation

Given a device placement, a graph is derived from the vertical and horizontal channels separating the device blocks. Nodes are placed at locations where ports are located and where horizontal and vertical routing regions meet. Any device placement topology may also be used (e.g., Fig. 5.1); however, we assume a channel-based placement is used. In a channel-based placement, such as depicted in Fig. 5.2a, nodes and edges are first derived for the vertical channels from the location of device ports and horizontal channels. These channels are then connected to other channels via horizontal interchannel edges, such as depicted in Fig. 5.2b.

Routes are selected by passing the routing graph and net-endpoints to a k-shortest loopless-route algorithm such as [97]. The set of available routes for a given net is selected from the top k-shortest routes; these routes are then analyzed.

5.3.5.2 Static route costs

Static route costs $\alpha^i_{k,static}$ are derived from the set of edges a route traverses E^i_k, comprising a sum of edge cost weights as in (5.10) and the number of bends traversed.

Though straight-waveguide losses at these scales are negligibly small, longer routes have a greater potential for intersecting other routes, potentially causing more crossings. Edges are therefore weighted according to their length in the substrate. Waveguide bends also have a cost associated with them as they can be a significant loss mechanism. In order to penalize their use, we modify the graph by adding weighted *transition edges* (via edges) connecting between vertical and horizontal nodes, as depicted in Fig. 5.1 and Fig. 5.2b. With all graph edge costs α^e_{cost} precomputed as weights, the overall static cost is a simple sum:

$$\alpha^i_{k,static} = \sum_{e\ =\ \text{edge of } r^i_k} \alpha^e_{cost} \tag{5.10}$$

5.3.6 Interroute Losses

Interroute losses $\alpha^{i,j}_{k,l}$ are caused by waveguide crossings. The fact that routes share graph edges does not necessarily imply that a crossing will be required. Consider two nets p and q in Fig. 5.3, where net q has two possible routes: $q(1)$ and $q(2)$. While both routes of q share routing edges with p (represented as routing regions in the diagram), only route $q(1)$ requires a crossing; route $q(2)$ does not. The technique only accounts for *required* crossings for interroute costs. Assuming no additional weighting, route $q(1)$ would have an interroute loss of one (1), whereas route $q(2)$ would have zero (0).

Determining a required crossing between two routes depends on the endpoints of a shared path. Consider the two nets depicted in Fig. 5.4a, where route A and B share edges in the middle. At the endpoints of the shared edges, the two routes diverge; we denote these as diverging endpoint edges (DEEs). By analyzing the relative directions of these DEEs, we can determine whether a crossing is required.

We introduce the concept of *rotation* to determine whether a crossing is produced by a shared path. For a given endpoint, *rotation* is the direction a route's DEE must rotate towards DEE of the other route, pivoted on their shared node, on the arc that does not contain the shared route edges. For example, in the left path endpoint of Fig. 5.4a, the DEE of A at (1) rotates counter-clockwise

towards the DEE of B. Likewise, at (2), the DEE of A again rotates counter-clockwise towards the DEE of B.

Proposition 5.1 [Crossing rotation pairs] *For a given path, with two route-endpoint DEEs x and y and respective rotations R_x and R_y (clockwise or counter-clockwise), a crossing is only* required *if* $R_x = R_y$.

Connected DEE-pairs of any angle (e.g., 180 degrees) are considered functionally equivalent 90° angled DEE-pairs retaining the same rotation relationship; this is depicted in Fig. 5.4b. A crossing, if it is required, will occur only once for a given shared path; we can treat the shared path edges as a single node that retains the crossing of the original.

These two transformations are combined by connecting the pivots of the DEE-pairs, and comparing two cases: 1) where the rotations of both endpoints are the same (rotating from A to B), and 2) when the rotations of both endpoints are different. Figure 5.5a clearly demonstrates how the same rotation forces DEEs of a given route to reside on opposite sides of the other route—inducing a crossing. When rotations are opposite, as in Fig. 5.5b, a given route's DEEs are adjacent and can connect together without requiring a crossing.

Exchanging rotation directions for the above cases covers all additional rotation cases, and therefore DEE-pairs with the same rotations will always induce a crossing, while opposite rotations will not.

Using Proposition 5.1, let i and j be two nets each with respective routes k and l and let P be the set of all paths shared by k and l. The value of $\alpha_{k,l}^{i,j}$ is computed as

$$\sum_{p \in P} \begin{cases} 1 & R_x^p = R_y^p \\ 0 & \text{otherwise} \end{cases} \tag{5.11}$$

where, for a given path p, $R_{x/y}^p$ is the rotation direction of the first and second endpoints, denoted x and y respectively.

The presence of interroute loss can affect routing more than route length, forcing longer routes to be exercised. Consider the nets in example Fig. 5.3, where a net q can utilize one of two distinct routes (1) and (2). Route (1), though shorter than route (2), must cross the chosen route for p; to avoid the crossing, route (2) could be utilized. Route (2), however, crosses over the chosen route for r. Should route r have less stringent loss constraints than route p, route (2) may be chosen over (1), despite a longer overall path. Ultimately, the final route choice is derived from a combination of all loss factors.

5.4 Results

The global routing formulation is applied to a number of optical routing problems. These problems were produced using the optical device placement methodology described in the previous chapter where devices are arranged into rows in the manner of standard cells. A routing grid is derived from the placement of devices and the location of device-ports. For each route, a number of candidate routes are generated, and the routes are analyzed for equation weighting. The router also assigns capacity constraints to grid edges based on the routing-area margins of the device blocks within the rows. The graph and set of candidate routes are then translated into an MILP problem formulation. The MILP problems are solved by the commercial optimization tool Gurobi [98]. Solutions to these MILP problems are then remapped onto the routing grid for visual inspection.

Figure 5.6a depicts the congestion map of a small optical logic design. Areas of higher congestion along edges are depicted with lighter colors (e.g., white or yellow) than less congested areas (e.g., black or brown). Highly congested zones are centered in areas where well-connected logic blocks are clustered together. Capacity constraints ensure that no capacity overflow is present. Larger designs such as Fig. 5.6b show the same pattern of congestion related to logic clustering.

5.4.1 Global Routing in a Full Adder Design

Our 1-bit full adder design also requires global routing to determine interchannel waveguide routes. These are depicted in Fig. 4.1. The routing is based on the network depicted in Fig. 3.15, where ring-resonators are used as the switching element.

5.5 Conclusion

This chapter presents a crossing-aware global routing model for integrated optics. The MILP model optimizes for crossings and bends and also limits congestion in the routing solution. We have demonstrated the application of this model to a number of optical routing problems and produced usable routing solutions in the context of crossing minimization.

We wish to emphasize that contemporary VLSI routers are *very* sophisticated, with the ability to route for complex designs well beyond the optical designs presented in this research. The main routing contribution for this dissertation is in *detailed routing*—the subject of the following chapter. However, in order to perform detailed routing, a crossing-aware global routing solution is required. For VLSI routers, crossing minimization at a global level is not incorporated. Therefore, instead of utilizing VLSI-centric global routers, we developed our own to create usable designs for detailed routing.

Once these designs are produced, we move onto detailed routing. In essence, the global router presented in this chapter is a tool development born out of necessity. Our router achieves its purpose

and incorporates many of the constraints and concepts present in VLSI routers. However, it is neither a sufficiently fine-tuned router, nor meant to compete with contemporary routing tools in the VLSI domain.

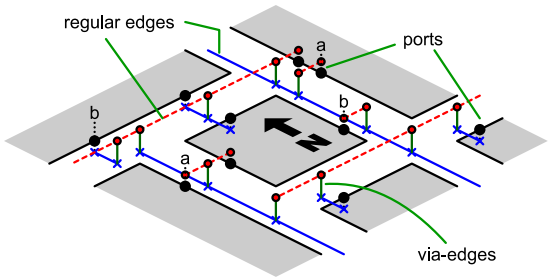


Fig. 5.1 Routing graph derived from device placement

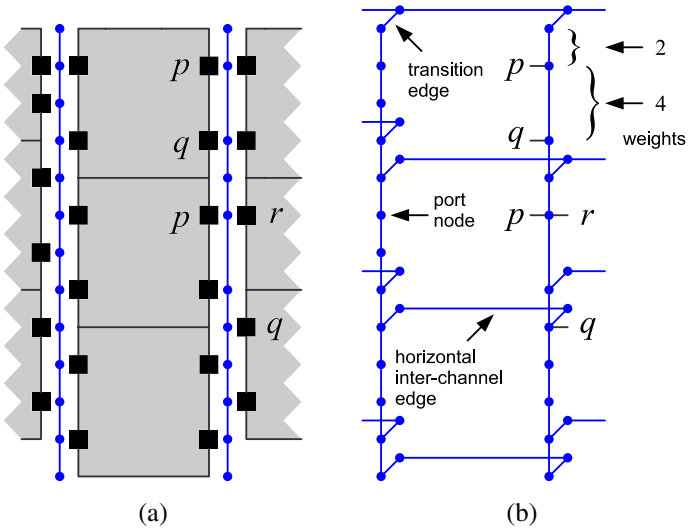


Fig. 5.2 Construction of routing graph from channel layout: (a) Vertical nodes and edges from layout (b) Complete routing graph

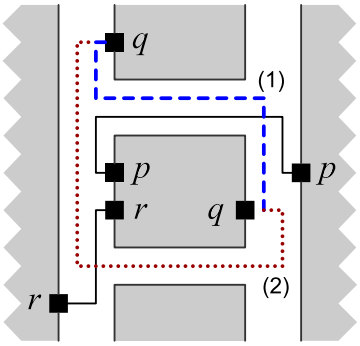


Fig. 5.3 Different route choices inducing different crossings

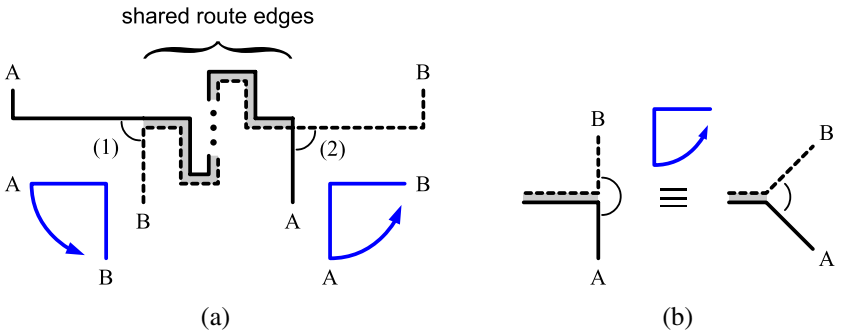


Fig. 5.4 Rotation of A to B at the endpoints of a shared path: (a) Conditions for determining route crossings (b) Rotation-equivalent routings

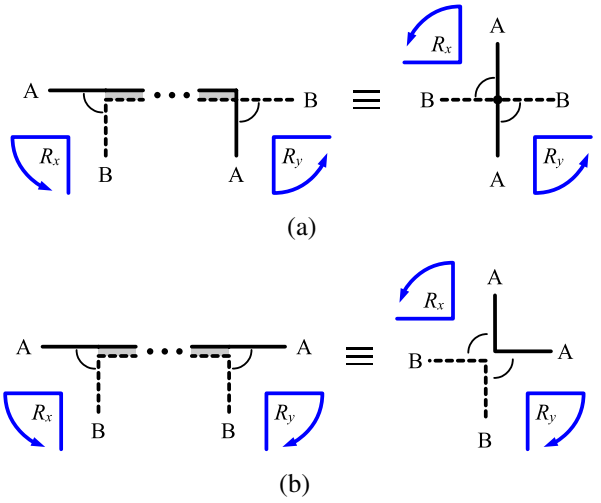
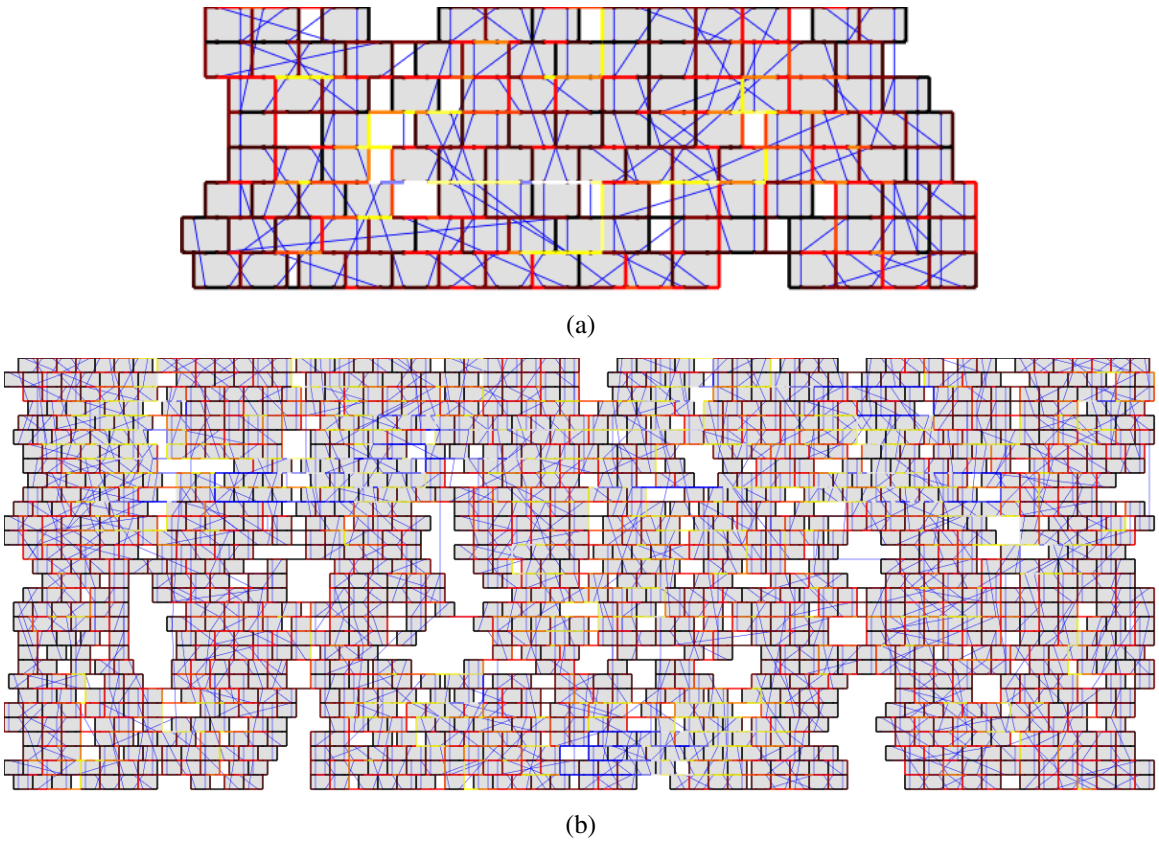


Fig. 5.5 Functionally equivalent configurations of path-endpoints and their rotations: (a) Same direction ($A \rightarrow B$) induce crossings (b) Opposite directions require no crossings



(a)

(b)

Fig. 5.6 Congestion maps of two routed designs: (a) Congestion map of small routing benchmark (b) Congestion map of larger routed benchmark

CHAPTER 6

CHANNEL ROUTING FOR INTEGRATED OPTICS

This chapter presents a methodology and solutions for detailed routing of integrated optical waveguides. In particular, we show that the detailed routing problem manifests itself as a *channel routing* problem, where (Silicon) optical waveguides are fabricated on a planar substrate and are connected to devices at the ends of the channel.[1] Planar routes require waveguides to bend (curve) and cross each other—causing loss of signal power. Channel routing techniques are therefore needed that minimize waveguide crossings and bends. This work presents two techniques: 1) a channel router based on sorting and 2) a channel router based on crossing-aware, graph-constraint track-assignment. Both routers minimize signal loss as a function of waveguide crossings and bends within the channel while also reducing area.

6.1 Problem Formulation

This chapter is concerned with channel routing and not with device placement. It is assumed that a (column-based) placement of optical devices is already given, along with the general routing path/topology of optical signals. This, subsequently, gives rise to a channel routing problem—such as the one redepicted in Fig. 6.1—which we solve while minimizing signal loss.

The waveguide connecting two ports is denoted as a *net* and comprises a single route with no signal sharing (fanout). Signal sharing is explicitly provided by predesigned waveguide *splitter* devices; these devices are treated as placed, 3-port, predesigned optical devices, with ports for routing. Therefore, our methodology renders every net a *two-terminal net* within the channel.

Work in optical interlayer connections exists [99], [100]; however, demonstrated interlayer connections come with a high penalty in terms of manufacturability—such as precisely aligned mirrors and grating couplers—and ultimately signal loss (2.5–3dB/connection). Such frameworks

[1] In the VLSI domain, channel routing is no more a topic of extensive research investigations due to the availability of a large number of metal layers and over-the-cell routing. This work revisits channel routing specifically for optical technology, which introduces new optimization criteria not addressed by VLSI channel routers.

are best suited for extraordinary routing in the presence of multiple optical routing layers, should they exist. Minimizing losses within each optical layer is therefore imperative.

We begin with an overview of signal loss mechanisms on which optimizations are based. Following is a description of the contributions of this work along with previous work in integrated optic design. We then describe our investigations into existing implementations of non-Manhattan grid sorting routers, the limitations of which lead to the development of our own sorting-based router. After describing our sorting-based router, we investigate a second channel router: a crossing-aware, left-edge style router. We then compare and evaluate the performance of our channel routers on a number of optical design benchmarks. The chapter concludes with an analysis of the results and concluding remarks.

6.1.1 Optimization Objective

The primary optimization objective in our routing formulation is *signal loss* minimization. Within the channel, this is achieved by 1) minimization of the total number of waveguide crossings and 2) minimization of the number of waveguide bends. Minimization of the number of tracks (channel height) is the subsequent secondary objective.

We optimize for the *total signal loss within the channel* due to optical feedback within the system. For example, in the 1-bit full-adder circuit depicted in Fig. 4.1 a signal may be routed such that it enters a given channel multiple times and may cross multiple other nets; this is depicted in the highlighted signal path. Therefore, instead of minimizing losses on a per-net basis, we minimize for *total* losses within a channel.

6.1.2 Contributions of This Work

This chapter presents methods for channel routing of integrated optical waveguides fabricated on a planar substrate. Two distinct crossing-aware channel routing techniques are presented: 1) a sorting-based router based on a non-Manhattan routing grid and 2) a left-edge style router utilizing crossing-aware graph-constraints. Both techniques are crossing-minimal and are constrained in a technology-suitable fashion to reduce bend-loss and area (number of tracks).

Our sorting-based channel routing technique utilizes a non-Manhattan routing grid and positional net sorting. We draw inspirations from sorting-based routers [101], [102]. These routers have useful properties of being *minimal in terms of crossings*. In our investigations, however, the original sort-router formulation suffers from a number of unaddressed limitations and detrimental side-effects that make it impractical. We show that there are fundamental flaws in the way the original swap/sort-routing channel problems are encoded, requiring excessive area, introducing more waveguide bends, and even affecting the original problem specification.

This leads us to develop our own crossing-minimal, bend-reducing, sorting-based router. We overcome the problems of the original formulation by 1) performing routing separately on both sides of the channel and 2) constraining the formulation to avoid unnecessary bends and enable routes to better utilize the routing grid. As a result, our router not only retains minimal crossings, but further minimizes the number of waveguide bends. Track utilization is also greatly improved over the original technique.

We also present a channel router based on traditional, *left-edge style* constraint-graph track assignments [48]–[50]. For such channel routing, we show how 1) crossing constraints are incorporated into the underlying constraint model enabling the routing solutions to be both crossing-minimal and 2) exploit the physical realization of waveguide curves for improved track utilization. For the former, the concept of *pin-rotation* is introduced as a means to determine whether nets require crossings. For the latter, we utilize the geometric properties of waveguide curves to enable knock-knees (KK) to facilitate track sharing, reducing overall track height.

Our channel routing techniques are then applied to a number of optical waveguide routing problems derived from photonic designs. We evaluate and compare the techniques with respect to each other in terms of crossings, bend-loss, and channel-height.

6.1.3 Previous Work

In VLSI physical design, channel routing algorithms [48]–[50] are textbook knowledge [45], [103]. Crossing minimization in routing has been studied in the context of the crossing distribution problem (CDP) [104], [105]. The CDP is concerned with the distribution of a minimal set of crossings within a routing topology; this is performed through permutations of net orderings. In contrast, while our work also utilizes net orderings to ensure crossing minimality, the final routing is performed with the goal of reducing *signal loss* with respect to crossings and bends—not the *distribution* of crossings across a layout.

Channel routing with crossing minimization has also been studied in the context of QCA routing [106]. Track assignments for multiterminal nets induce varying numbers of crossings; therefore, [106] formulates crossing minimization, heuristically, as a weighted-minimum-feedback-edge-set problem. However, in the context of our problem—utilizing exclusively 2-terminal nets—we exactly minimize waveguide crossings, obviating the need for such an approach.

6.2 Non-Manhattan Grid, Sorting-Based Routing

A channel routing problem is represented by net "pins" fixed to the top and bottom of a channel. The purpose of the router is to route all nets in the channel's routing region while minimizing parameters such as area (channel height, number of tracks), signal delay (net-length), or signal loss.

Figure 6.2 depicts a minimum track channel routing obtained by a left-edge router. In traditional VLSI channel routing, area and net-length are primary optimization goals. Channel routing also seeks to minimize other objectives, such as vias [107], and in the case of optics technology, signal loss.

Manhattan-based (rectilinear) grids are traditionally employed in VLSI routing, dedicating layers specifically to horizontal or vertical spans for routing flexibility; non-Manhattan-based grids (e.g., octilinear) are rarely utilized. Integrated optic waveguides are, however, well suited to non-Manhattan-based routing grids. Such routing grids can suitably represent waveguide curves and provide greater routing flexibility in the absence of multiple routing layers.

The work of [101], [102], also described in textbook [103], investigates a non-Manhattan grid channel router based on *sorting*. The nets of a channel are assigned numerical indices, and routing is performed by sorting the nets in a finite number of permutations. The number of permutations performed represents the number of tracks utilized. Examples of this sorting-based routing are depicted in Fig. 6.3.

6.2.1 Crossing Minimality

In addition to utilizing non-Manhattan grids, sort-router's channel solutions are minimal in terms of the number of crossings. Crossing minimality results from the fact that [101]: 1) crossings only occur if nets are positioned out-of-order during sorting and 2) once sorted, pairs of nets *never cross each other again during the sorting process.*

The flexibility of a non-Manhattan-based grid and crossing minimality makes sorting-based routing attractive for integrated optics. However, the original sorting-based channel routing solutions presented in [101], [102] have drawbacks that make them impractical. Below, we describe the limitations of the sort routers of [101], [102]; these limitations motivate the design of our own sorting-based channel router, specifically designed for integrated optics.

6.2.2 Sorting-Based Channel Routing

Two sorting techniques are presented in [101], [102]: swap-sorting and bubble-sorting. The swap-based router swaps positions of pairs of adjacent nets if they are out-of-order. For example, in the bottom track of Fig. 6.3a, nets 5 and 4 are out-of-order, and they swap columns in the transition to track 1; this is reflected in the swap depicted in Fig. 6.3b. A bubble-sort based technique can also be used, as depicted in Fig. 6.3c, allowing indices to sort across multiple column positions. Bubble-sorting, however, causes nets to cross at 45° relative angles and therefore is *unusable* for optical waveguide routing. *Our channel routing technique utilizes swap-based sorting for routing.*

6.2.3 Encoding Side-Only Nets

The described channel problem setup assumes that all nets appearing on the bottom of a channel also appear on the top. However, most channel problem instances also incorporate nets with pins exclusively on one side of the channel. Also, empty spaces *between* pins (i.e., "gaps") or within the routing tracks are also not accounted for by the basic routing algorithm. We first formally define net types with respect to their pin locations as well as the concept of empty spaces in the routing region:

Definition 6.1 [X-net, T-net, B-net, side-only nets] *A net with pins on both the top and bottom of a channel is an **X-net** (cross-over/shared net). Nets with pins exclusively on one side of a channel are denoted **side-only nets**: a **T-net** (top-only net) has pins on only the top side of the channel; a **B-net** (bottom-only net) is a net with pins on only the bottom of the channel.*

Definition 6.2 [Gap] *A **gap** is an empty location in the routing grid, or an empty pin location. Gaps are not assigned sorting indices, but may be routed over if unoccupied.*

Consider the channel depicted in Fig. 6.4. We must assign net pins initial sorting indices in order to route the channel nets. Observe nets E, B, and D. Net E is an X-net, and therefore the bottom pin of E is assigned the index corresponding to its top pin's index-position, in this case 4. Nets B and D are B-net and T-net types, respectively; they have their pins exclusively on one side of the channel—*side-only nets*. The work of [101] provides a means for encoding T-nets and B-nets into the channel sorting problem:

6.2.4 B-net Encoding

For a given B-net, the left-side pin is assigned a high-valued index, and the right-side pin is assigned a negative index. For example, in Fig. 6.4, the left-pin of B is assigned index 13—*a value greater than the number of channel columns*—and the right-pin is assigned -2. The result of these index assignments is that the high-valued index causes the route of the left-pin to sort to the right; the negative index causes the route of the right-pin to sort to the left. When the routes meet, the net is considered routed and the indices are removed from the problem in subsequent tracks.

6.2.5 T-net Encoding

T-nets do not exist at all on the bottom track and therefore are added to the initial bottom sorting track as *additional columns ("virtual columns")* to the left and right of the original channel columns. In Fig. 6.4, these are indicated by the shaded areas. The bottom pins of the T-nets are assigned indices corresponding to the positions of the top-pins of the net; this causes the respective routes to sort towards these positions during the sorting. The routes of the T-nets must, however, meet at

some point within the channel. To facilitate this, T-net pins are assigned to the sides *opposite* of their relative positions in the top track. This causes the routes to cross each other at some point on their way to their final positions.

In Fig. 6.4, net *D*—a T-net—has pins on top of the channel at columns 3 and 9. These column positions are used as the indices assigned to the pins at the bottom of the channel—index 9 on the left, 3 on the right. As the sorting proceeds, the routes for the T-nets converge towards each other, cross at some point within the channel, and continue to their final sorted position. After the channel routing completes, only the routing above the crossing point of a T-net is retained as the actual T-net route (e.g., the solid-line route of *D* in Fig. 6.4).

6.2.6 Limitations of Swap Router

Consider the swap-router solution depicted in Fig. 6.4. Immediately apparent are the following problems:

1. wasted tracks (gaps) with suboptimal routes;
2. the final (top) positions of routed net terminals are shifted to the right, as compared to the original specification;
3. routes detour away from destination column position, creating more "bends," which can cause optical signal loss;
4. routes exist outside the channel's column bounds.

These problems arise from empty spaces (i.e., "gaps") in the channel problem and the encoding of T-nets.

6.2.7 Gaps in the Sorting Problem

Gaps in the channel problem affect the relative positioning of nets in a track while not providing any sorting information themselves. The end result is that the routing solution only respects the *relative* positioning of nets, not the *absolute* position. This is problematic as the routed nets of the top track may not reflect the positions of the pins in the original specification; this is demonstrated in the top track of Fig. 6.4. Both B-nets and T-nets introduce gaps into the sorting problem.

6.2.8 T-net Encoding

As demonstrated in Fig. 6.4, the encoding for T-nets *negatively impacts virtually all aspects of the channel solution.* T-nets introduce additional channel columns that affect positioning of the net pins as well as enabling routes to extend outside the channel's original column bounds. The T-net temporary/virtual routes (dashed lines in Fig. 6.4) also cause unnecessary detours for any other route they interact with, increasing track count—a track for each crossing—during the sorting. For

example, in Fig. 6.4, the T-net route originating from the left-side for net J must cross seven (7) other routes to form a solution. None of the temporary routing below the cross-over point serves any real purpose in the final channel solution. This also causes other routes to move outwards, such as the route for E detouring left as it crosses the left terminal route for A, D, and J.

6.2.9 Postprocessing

Accepting the solution from the swap router as-is would require an additional postprocessing step to 1) position T-net terminals correctly with respect to the original specification, and 2) postprocess routes within the solution to improve track usage and prevent routes from extending outside the bounds of the channel. While some postprocessing work is performed in [102] to compact the track space, gap handling remains unaddressed. Larger problems, especially with many T-nets, can produce extremely large solutions, most of which is wasted space. This effectively defeats the purpose of utilizing the swap router in the first place.

6.3 2-Sided Swap Routing (2Swap)

To overcome the limitations of the original sorting routing, we introduce the 2-sided Swap Router (2Swap): a sorting-based router that performs routing from both sides of the channel simultaneously. Sorting still remains a key component of routing, ensuring crossing-minimality; however, the routing from both sides overcomes two key limitations of the original router:

- The elimination of T-nets from the routing solution. No additional columns are added to the channel problem and no temporary routes are needed.
- Sorting in the presence of gaps is addressed, and pins on each side of the channel are fixed as per the original specification.

As in the original swap router, the 2Swap router produces a crossing-minimal solution. This is ensured by updating the position of route pins at each iteration of the routing. A swap that takes place on one side of the channel is reflected when the other side is routed, retaining the sorting-based assurance of crossing-minimality.

6.3.1 Gap Crossing

Gaps are an intrinsic part of virtually every channel routing instance. The presence of gaps, however, is not even mentioned by [101], and other examples in literature [102] only depict and describe dense (gapless) routing problems such as Fig. 6.3. We address the existence of gaps within the sorting problem by allowing routes to span horizontally across gaps to the point of crossing. During sorting, pairs of routes are analyzed across gaps. Should a swap be possible, a horizontal

span is created up to the point of crossing, and the swap then occurs. This is depicted in Fig. 6.5c, where a route for net B traverses multiple gaps ($\emptyset$) to cross over route A on the right.

6.3.2 Two-Sided Swap-Routing

The 2Swap router alternates between both sides of the channel for swap-routing while performing bookkeeping to ensure that crossings are not performed twice and sorting-indexes are updated. We now define two different net-concepts:

Definition 6.3 [Source/destination sides, S-net, D-net] *Given a channel side being routed, routing is performed from the **source** (src) to **destination** (dest) sides of the channel. The src/dest sides of the channel are analogous to the bottom/top sides when routing is performed from the bottom-to-top. Given a channel side being routed, an **S-net** is a **source-side net**—a net with both pins exclusively on the source side of the channel. Likewise, a **D-net** is a **destination-side net**—a net with both pins exclusively on the destination side of the channel.*

The technique works in the following manner:

1. For each iteration of the 2Swap router, swap-routing is performed for a single track on both sides of the channel, subject to the following conditions:
   - For each iteration of the sorting, relative ordering of nets is recomputed for swap-routing, utilizing the current set of sorted tracks. This helps ensure that the routing performed for a given side is aware of net-swaps that took place on the opposite side of the channel.
   - For a given side of a channel being routed, only S-nets and X-nets are encoded as indexes and routed. D-nets are treated as gaps on the destination track for purposes of routing. This ensures that D-nets are only routed on their respective side as S-nets, avoiding the problems previously associated with T-nets in the original sorting-based router.
   - Any S-nets that have completed routing are removed in subsequent tracks (replaced by gaps).
2. Routing completes when routes on both the top and bottom are completely sorted with respect to each other.
3. A postsort routing is performed to provide a routable channel.

The 2Swap technique is given in Algorithm 3.

6.3.3 Postsort Routing

The sorting phase of routing completes when all S-nets are routed, and all X-net-indexes are sorted with respect to each other. The latter condition, however, does not guarantee that the channel

is fully routed. Consider the 2Swap routing solution depicted in Fig. 6.5a. While the X-nets in the middle of the channel are *sorted* with respect to each other, they are not fully *routed* as their column positions are misaligned. In such cases, *postsort routing* must be applied, as demonstrated in Fig. 6.5b.

6.3.4 Solution Quality

We route the channel depicted in Fig. 6.4 using the new 2Swap router and observe that it produces a far more usable solution as depicted in Fig. 6.6a. Fewer tracks are utilized (6 vs. 11), and no additional columns are added to the routing solution. In addition, the top and bottom pins' locations are the same as in the specification. Overall, the solution produced by the 2Swap router is improved with respect to the original swap-router. However, further improvements are still possible, especially in terms of waveguide bends.

6.3.4.1 Excess Bends and Their Causes

The 2Swap route produces a large number of excess bends in its solutions. This is especially apparent in larger channel instances such as Fig. 6.5b. The excess bends in 2Swap routes are due to the fact that while the position of net *terminals* are fixed to *absolute* positions, the intermediate sorting is still a *relative* sorting-position operation. Pairs of routes for a given net are therefore not actively converging towards each other during the sorting. For example, in Fig. 6.6a, net E swaps with net B, shifting E to the right—in the opposite direction of its destination pin. As a result, route E must detour across a large expanse of space to connect both sides—despite being only one column away in the original problem. The same problem afflicts net I, which detours left in its swap with H.

As a signal loss mechanism, bends must also be accounted for, and we address these with a *constrained* 2-sided Swap Router.

6.4 Constrained 2-Sided Swap Routing

The Constrained 2-Sided Swap Router (2SwapC) introduces the following key features:

- **Convergence and swapping constraints:** Routes cannot change columns unless swapping or utilizing a horizontal span. Swapping only occurs between adjacent routes, under the condition that the swap results in each respective route converging towards its respective opposite pin.
- **Horizontal spanning:** Routes may now cross each other using horizontal spans in addition to diagonal (swap) crossings. For a given route, the horizontal spans are only permitted as a direct connection to the paired pin's route, completing the routing.

The 2SwapC technique is given in Algorithm 4.

6.4.1 Convergence Constraint and Swapping Restrictions

Routes are restricted from shifting columns, except during a swap or utilizing a horizontal span (explained later). Swapping is still allowed, but restricted: adjacent net-routes must both converge towards their opposing pin. The convergence swapping constraint work together to reduce the number of bends in the routed solution.

Consider the channel routing solution depicted in Fig. 6.7a where the convergence constraint only restricts routes from routing horizontally *away* from the opposing pin. The expectation of this convergence constraint is that by converging horizontally, the routing would reduce area. The reality, however, is that horizontal motion does little to reduce track height. Instead, routing solutions suffer from large numbers of unnecessary bends as the routes aggressively follow the contour of adjacent routes.

To reduce unnecessary bends, we strictly constrain routes to only shift horizontally during a swap. In the absence of such a swap, this restriction results in straight vertical connections, as depicted in Fig. 6.7b. While the routes cover more horizontal area, that area is already empty. In addition, the overall track height generally remains the same or is often reduced, as will be seen later.

The convergence constraint restricts the swap-sorting of the original routing technique. For example, consider the channel problem instance in Fig. 6.8(a–b). In the first iteration, routes move in the direction of their paired-routes, e.g., B and C swap on the bottom, and the top-route of C moves left. In the second iteration, Fig. 6.8a, the bottom routes of A and C cannot swap; this would violate the convergence constraint by forcing C to the right. The bottom route for A can only move vertically. At this point routing stops, leaving the solution in an incomplete state. We therefore introduce our second extension: horizontal spanning across vertical routes.

6.4.2 Horizontal Spans and Crossings

In Fig. 6.8a, the bottom route of A is *blocked* by C due to the convergence constraint. To complete routing, such blocked routes must be allowed to cross each other without swapping. We achieve this by allowing routes to cross using horizontal/vertical crossings (HV-crossings) under certain conditions.

Vertical spans form naturally due to the convergence constraint as depicted in Fig. 6.8b. As vertical spans form, a horizontal span is allowed in cases where crossing other routes does not result in a sorting violation. Horizontal spans are also only allowed to cross vertical spans that traverse two (2) or more tracks, to ensure that bends will not occur at the junction. This is depicted in

Fig. 6.8b, where the horizontal crossing may not cross at track 1, but must instead cross at track 2. Furthermore, to reduce the number of bends, routes may span horizontally only if they can make a *direct* connection to the column of their respective paired route. The effect of this is that any given net will only make one horizontal span within the channel. This can be observed in the channel solution depicted in Fig. 6.7a.

6.4.3 Comparison with 2Swap

While the number of crossings is the same, channel solutions produced by 2SwapC have fewer bends than 2Swap. This is evident in Fig. 6.6 where the 2Swap router produces many routes that "zig-zag" throughout the routing region, whereas 2SwapC utilizes straight routes with few or no bends. The net result is fewer bends, more vertical spans to permit horizontal cross-overs, and retaining the ability for neighboring nets to cross diagonally—reducing tracks. Moreover, the restrictions still retain the sorting mechanism of the other swap routers, ensuring crossing minimality.

The 2SwapC router is applied to a number of benchmarks later in this chapter to evaluate its performance. We also investigate incorporating crossing-aware constraints onto traditional left-edge style routers, which we describe in the following section.

6.5 Left-Edge-Style Channel Routing

Traditional left-edge-style channel routers [48]–[50] represent the channel routing problem using horizontal and vertical constraint graphs (HCG, VCG). An alternate representation of the HCG is the zone representation, which is derived from the HCG, where every zone is defined by a maximal clique. The number of signals in the largest zone is the lower bound on the number of tracks needed for routing. These graphs encode constraints on how tracks may be assigned to nets in the channel. Consider the channel routing problem depicted in Fig. 6.9a. The resulting zone representation is depicted in Fig. 6.9b. Likewise, the VCG for the problem is represented in Fig. 6.10a.

A net may be assigned to a track should it have no descendants on the VCG and have no overlapping zone conflicts with previously assigned nets on a given track. Nets are removed from the VCG as they are assigned to tracks. When a track cannot contain more nets, a new track is created and the process is repeated until no more nets are left for assignment.

Multiple nets can be candidates for assignment to a given track, each with different horizontal overlaps. Therefore, heuristics are used to choose which nets are assigned first. One of the simplest is a greedy heuristic used in *constrained left-edge channel routing* [48], where the left-most available nets in channel are assigned first to tracks. This can lead to suboptimal track-utilization; more

sophisticated heuristics analyze the graph structure for better results, such as [50], which attempts to reduce the longest path in the VCG for better track utilization. We refer to the class of track assignment algorithms above generically as "left-edge-style" channel routing. The approach we describe below can be incorporated into any such techniques.

6.5.1 Crossing-Constrained Track Assignment

Figure 6.11a depicts the output of a (VLSI) left-edge 2-layer channel router, and Fig. 6.11b, a channel routing constrained for crossing-minimization. Both solutions are minimal in terms of tracks; however, the total number of crossings in Fig. 6.11a is 10, compared to 8 in Fig. 6.11b. The discrepancy in the number of crossings is attributed to the two crossing points caused by nets B and C. By forcing C to appear below B, two crossings are avoided. However, transforming from Fig. 6.11a to Fig. 6.11b is not as simple as moving net C below B, not if track height is to be kept minimal. Crossing minimization must therefore be encoded into the routing process itself as constraints.

We constrain the channel routing problem to favor crossing minimization. The VCG is modified such that avoidable crossings impose vertical constraints on the net ordering. Only nets that share zones have the possibility of crossing, and pairwise analysis takes place after the zones are derived.

A crossing constraint is only encoded into the VCG if a crossing can be avoided. For example, the pair of nets in Fig. 6.12c would not normally be constrained in the VCG; however, a net crossing can be avoided if B is assigned a track above A. Therefore, an edge connecting B to A is added to the VCG. Conversely, the two nets in Fig. 6.12b cannot avoid crossing, and therefore no constraint is added.

We introduce the concept of *pin-rotation* to detect avoidable crossings. If we were to map the pins of nets on a unit circle, a crossing is unavoidable if rotating from one pin to the next is not possible without first passing through the pin of another net. Consider the nets depicted in Fig. 6.12a. Collapsing the shared horizontal region and considering the areas Fig. 6.12a(1) and Fig. 6.12a(2) shows how pins of a given side rotate with respect to each other (clockwise/counter-clockwise) around an axis fixed at the center. In the case of Fig. 6.12a(1), the rotation of the left pin of A to the left pin of B is *counterclockwise,* and likewise the pins on the right-side also rotate in the same *counterclockwise* direction. If the pins on both left and right terminals rotate in the same direction, a crossing is unavoidable. More formally

$$X_{A,B,CW}^{left} = \begin{cases} X_{B,top}^{left} & \textbf{if } C_A^{left} < C_B^{left} \\ \neg X_{A,top}^{left} & \text{otherwise} \end{cases} \tag{6.1}$$

$$X_{A,B,CW}^{right} = \begin{cases} X_{A,top}^{right} & \textbf{if } C_A^{right} < C_B^{right} \\ \neg X_{B,top}^{right} & \text{otherwise} \end{cases} \tag{6.2}$$

$$X_{\text{avoidable}}(A,B) = \left(X_{A,B,CW}^{left} \neq X_{A,B,CW}^{right} \right) \tag{6.3}$$

where $C_N^{left/right}$ is the integer-valued column-position of a pin of net N on a given side (left, right); the Boolean variable $X_{N,top}^{left/right}$, using the same notation, denotes whether that pin resides on the top side of the channel. Equation (6.1) and (6.2) utilize the horizontal relationships of pins and their channel-sides (top/bottom) to determine the *clockwise rotation* (CW) of a given pair of $left$ or *right* pins for nets A and B, rotating from A to B. A crossing is avoidable only if left and right rotations are *not* the same, the result of (6.3).

For example, in Fig. 6.12a, consider the left side of the shared span indicated by (1) in Fig. 6.12a:

- The variables C_A^{left} and C_B^{left} are the column positions of the respective left-terminals of nets A and B. In the example, $C_A^{left} = 1$, $C_B^{left} = 2$.
- $C_A^{left} < C_B^{left}$ implies $X_{A,B,CW}^{left} = X_{B,top}^{left}$ from (6.1).
- The left pin of net B is *not* on the top side of the channel ($X_{B,top}^{left} = \textbf{false}$). Therefore, the left side of the pair of nets is *not* rotating clockwise from A to B, i.e., $X_{A,B,CW}^{left} = X_{B,top}^{left} = \textbf{false}$.
- On the right side of the shared span (2) in Fig. 6.12a, $C_A^{right} < C_B^{right}$. This condition implies that $X_{A,B,CW}^{right} = X_{A,top}^{right} = \textbf{false}$. The right side is therefore also *not* rotating clockwise from A to B.

Having the same direction of rotation ($X_{A,B,CW}^{left} = X_{A,B,CW}^{right} = \textbf{false}$) implies that a crossing is *unavoidable,* as determined by (6.3); this is reflected in the figure.

Applying crossing constraints to the problem depicted in Fig. 6.9a results in the VCG depicted Fig. 6.10b. As compared to the original VCG Fig. 6.10a, the crossing-constrained VCG is more heavily constrained, ensuring that unnecessary crossings do not occur, such as the double-crossing of nets B and C in Fig. 6.11a.

6.5.2 Knock-Knee Track Sharing

Though the modified VCG is effective in preventing waveguide crossings, the additional constraints can affect overall track height and may produce a worse solution in terms of number of tracks. However, we observe that the bend geometry of optical waveguides can be exploited to further reduce channel height. This is discussed below.

Consider the two nets in Fig. 6.13a. The endpoints of the two nets occupy the same column and therefore net A should be placed above B in the VCG. However, given the same track, the two

nets would intersect at a corner of each horizontal span—a *knock-knee*. In VLSI, this situation is untenable, and different tracks would need to be assigned to each net. However, for waveguides, the minimum grid spacing for a channel can permit knock-knees in the routing grid. This is depicted in Fig. 6.13a, where a track is shared between the two nets without overlap.

A knock-knee occurs where one net ends and another begins, e.g., nets C and E in Fig. 6.13c. During zone construction, at columns where knock-knees appear, the net that is beginning its horizontal span is only added to the *subsequent* column set, rather than the current column set under consideration. For example, in Fig. 6.13c knock-knee signals E, F, G, I, and J are removed from the marked columns and only appear in the subsequent columns.

The effect of this column change on the resulting zones is demonstrated in Fig. 6.13b, where there are six (6) zones rather than the five (5) from the previous zone analysis Fig. 6.9. Despite containing an additional zone, the largest column set now contains *one fewer* net than the original, resulting in the 4-track solution depicted in Fig. 6.13c.

Overall, the effect incorporating knock-knees into a routing solution is that two knock-knee nets can now occupy separate zones and therefore can be placed on the same track. Additional zones may be created; however, those zones are equal in size or *smaller* in terms of nets—*potentially reducing the lower bound on the number of tracks required for routing*.

6.5.3 Cycles Induced by Crossing Constraints

Crossing constraints can induce cycles in the VCG. Consider the three nets depicted in Fig. 6.14a. Without crossing constraints, nets A and B would be unconstrained, and no cycle would occur; however, due to the constraint edge between B and A, such a cycle occurs. Cyclic constraints cannot be routed without additional tracks and require "doglegging" to complete routing [49]. In order to avoid crossings, the routes for A and B are converted into doglegging routes as depicted in Fig. 6.14b, utilizing the same columns as the original. Unfortunately, this results in an additional two (2) tracks being added to the routing solution should spare tracks not be available adjacent to the cycle. However, in the presence of knock-knees, both the crossings, and the additional tracks can be avoided, as depicted in Fig. 6.14c. The experimental results show that knock-knees can have a marked difference in track utilization especially in the presence of cyclic constraints induced by crossings.

6.6 Experimental Results

We evaluate our sorting-based router (2SwapC) and our crossing-aware left-edge router (LE+KK) on a number of channel problem instances. We also compare against the Yoshimura-Kuh

(YK) router [50] as a baseline. The original sort-based router [101], [102] is not compared as it produces routing solutions that violate the original channel specifications.

Both 2SwapC and LE+KK routing techniques are implemented as compiled script-code. Problem instances incorporating as many as 512 nets were tested, and routing completed in under 30 seconds. Most routings complete in under a second.

6.6.1 Channel Problem Instances

In the VLSI community, it is customary to evaluate routing techniques against benchmark suites such as those described in [108]. However, such benchmarks are inapplicable to this work. The VLSI routing benchmarks utilize multiple routing layers, with nets incorporating large amounts of fan-out. This is due to the aggressive factorization-based logic decomposition/synthesis applied in the VLSI domain.

Our investigations found these techniques to be inapplicable to photonic logic. Subsequently, we proposed technology-specific netlist decomposition techniques specific to silicon photonics [2]. The channel routing instances are derived from relevant integrated optical designs.

Our channel problem instances are derived from the ACM/SIGDA (i.e., MCNC) logic synthesis benchmark suites [79]. These designs are synthesized into waveguide-connected, optical switching networks, utilizing our optical logic synthesis technique [2]. The optical netlist is then placed into rows using a VLSI row-placer [93]. Crossing-aware global routing is performed subsequently, utilizing an MILP-based approach on a set of candidate routes. After global routing completes, the regions between rows are extracted as channel problem instances. Multiple channel routing problems were derived using the above design flow.

6.6.2 Metrics

Channels are routed to evaluate their performance in terms of key metrics: 1) crossings, 2) bend-loss, and 3) track utilization (area). In all problem instances, the number of crossings produced by the 2SwapC and LE+KK routers is the same; that result is combined in Table 6.1. In our octilinear grid, bends can be either 90° or 135°, having different loss characteristics. Bend-loss α_{bend} is computed for each using (6.4) [109] as a function of radius of curvature (Fig. 6.15)

$$\alpha_{bend}(r) = C_1 \cdot \exp(-C_2 \cdot r) \tag{6.4}$$

where the constants C_1 and C_2 are dependent on the physical parameters of the waveguides. For simplicity and convenience, we use a unit grid for calculating bend-radius and select $C_1 = C_2 = 1$. This results in bends-loss for $\alpha_{bend}(\{135^\circ, 90^\circ\}) = \{0.135, 0.368\}$. In addition to counting bends

within the channel, we also include bends at the interface of the side of the channel to the first track of each respective side.

6.6.3 Analysis of Results

The two routers, 2SwapC and LE+KK, are tested on a variety of channel problem instances. The results of the routings are found in Table 6.1. Total time for both routers to complete all channel problem instances was less than 10 seconds. Analysis of the results reveals that

- **Crossings:** Both 2SwapC and LE+KK perform equally well in terms of number of crossings. Moreover, on average, both routers have 43% fewer crossings than the YK router.
- **Bend loss:** The LE+KK router produces nearly the same bend loss as YK router due to the similiar track-assignment routing formulation. However, 2SwapC consistently produces better results compared to LE+KK (5.6% less average bend loss).
- **Tracks:** LE+KK often utilizes fewer tracks than 2SwapC (8% less, on average). As expected, both 2SwapC and LE+KK produce more tracks than the strictly track-optimizing YK router, except in the two instances noted in the table. On average, the increase in tracks is 24% and 14% for 2SwapC and LE+KK, respectively.

In comparing 2SwapC and LE+KK, the 2SwapC router ultimately produces less bend loss. We attribute this to 2SwapC's use of a non-Manhattan grid, enabling waveguide bends with larger curvature (e.g., 135° curves). This contrasts with nets routed by LE+KK, which always require 90° bends. In terms of tracks, LE+KK offers slightly better track utilization due to enforced 90° curves and the single-track utilization model.

6.7 Conclusion

This chapter presents channel routing techniques for integrated optical waveguides fabricated on a planar substrate. We identify our primary optimization objective as signal loss minimization—in terms of waveguide crossings and bends-loss—with area as a secondary objective. We present two distinct crossing-aware channel routing techniques for integrated optics: 2SwapC—a sorting-based router based on a non-Manhattan routing grid, and LE+KK—a left-edge style router utilizing crossing-aware graph-constraints. Both techniques are crossing-minimal and are constrained in a technology-suitable fashion to reduce bend-loss.

Our new 2SwapC router addresses and overcomes the shortcomings of previous sort-based routers through the use of two-sided swap routing. We have further improved the router through additional routing constraints as well as innovative extensions that enable routes to utilize horizontal spans while reducing bends. Bend-losses are markedly improved as demonstrated in tests on many

channel routing instances, and 2SwapC demonstrates that it is superior in terms of bend-loss than the LE+KK router.

The LE+KK router we present builds upon traditional constraint-graph based ("left-edge style") routers to produce a crossing-minimal channel routing solution. This router incorporates additional constraints to ensure crossing minimality by analyzing pairs of nets using the concept of *pin rotation*. We also exploit waveguide curves to enable knock-knees to improve track utilization as well as to enable crossing-constrained doglegging, improving the solution quality.

Our channel routers provide effective means for automating optical waveguide routing with signal loss as a primary metric. When applied to channels derived from optical designs, 2SwapC and LE+KK both produce comparable results. In terms of our primary signal loss metric, however, we choose the 2SwapC router as its ability to utilize non-Manhattan grids gives it greater potential in reducing overall signal loss. The LE+KK router is still a good choice in cases where area is most important.

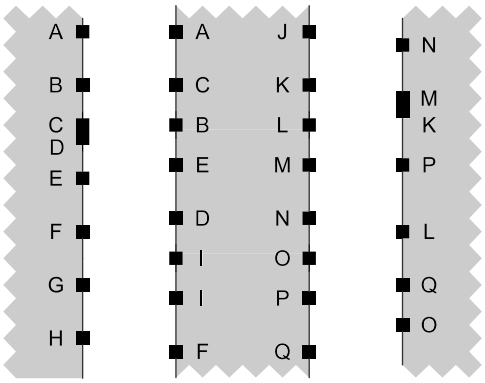


Fig. 6.1 Channels for detailed routing

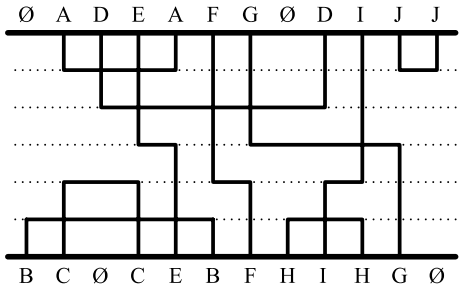


Fig. 6.2 Track-optimized channel solution

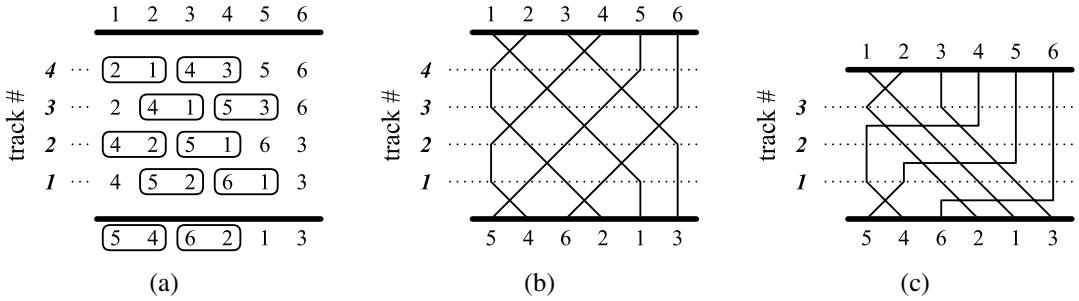


Fig. 6.3 Channel routing performed by sorting indices. Circled indices denote a pair that is reordered (sorted): (a) Swap-sorted net indices (b) Swap-sorted solution (c) Bubble-sorted solution.

Algorithm 3 2Swap: Main Swap Router

```
function TWOSWAPROUTER(channel)
    T_sol := [ ]        /* (solution tracks for channel) */
    repeat
        for each side S of the channel do
            T_src := copy of current track of src side
            T_dest := copy of current track of dest side
            Assign all D-net nets in T_dest to ∅
            T_unsorted := array of net-indexes of T_src with respect to T_dest
            T_sorted := SWAPSORT (T_unsorted)
            Add T_sorted to T_sol
            Remove completed S-nets from T_sorted
            Set current track for S to T_sorted
        end for
    until (all B-nets and T-nets are routed) and (all X-nets are sorted with respect to each other)
    return POSTSORTROUTING(T_sol)
end function


function SWAPSORT (T as array of indexes)
    for i = 1...sizeOf(T) do
        if T(i) ≠ ∅ and T(i+1) ≠ ∅ and T(i) > T(i+1) then
            SWAPVALUES(T(i), T(i+1))
            /* Additional increment to skip to next pair */
            i := i+1
        end if
    end for
    return T
end function
```

Algorithm 4 2SwapC: Main Swap Router

```
function TWOSWAPCONSTRAINEDROUTER(channel)
    T_sol := [ ]        /* (solution tracks for channel) */
    H_sol := [ ]        /* (horizontal spans for channel) */
    repeat
        for each side S of the channel do
            t := sizeOf(T_sol^src)
            T_src := T_sol^src[t]   /* copy of current track of src side */
            T_dest := T_sol^dest[t]   /* copy of current track of dest side */
            Assign all D-net nets in T_dest to ∅
            T_unsorted := net-indexes of T_src with respect to T_dest
            T_{y-1} := T_sol^src[t-1]   /* previously routed track */
            T_y := T_unsorted   /* currently routed track */
            /* next routed track: */
            T_{y+1} := CONSTRAINEDSWAPSORT (T_unsorted)
            H_spans := HORIZONTALSPANS (T_y, T_{y+1}, T_{y-1})
            Remove h-span-completed S-nets from T_{y+1}
            Add H_spans to H_sol;    Add T_{y+1} to T_sol
            Set current track for S to T_{y+1}
        end for
    until (all B-nets and T-nets are routed) and (all X-nets are sorted with respect to each other)
    return COMBINEDROUTINGSOLUTION(T_sol, H_sol)
end function
```

Algorithm 5 2SwapC: Swap sort and horizontal spanning functions.

function CONSTRAINEDSWAPSORT (T as array of indexes)
  for $i = 1 \ldots \text{sizeOf}(T)$ **do**
    if $T(i) = \emptyset$ **or** $T(i+1) = \emptyset$ **then**
      continue
    end if
    /* Swap constraints for $T(i)$ ***and*** $T(i+1)$ ****/***
    $x_i^a :=$ COLUMNOF($T(i)$)
    $x_i^b :=$ COLUMNOF(other route of $T(i)$)
    $x_{i+1}^a :=$ COLUMNOF($T(i+1)$)
    $x_{i+1}^b :=$ COLUMNOF(other route of $T(i+1)$)
    $cond_i := \begin{cases} \textbf{true} & (x_i^b - x_i^a) > 0 \\ \textbf{false} & \text{otherwise} \end{cases}$
    $cond_{i+1} := \begin{cases} \textbf{true} & (x_{i+1}^b - x_{i+1}^a) < 0 \\ \textbf{false} & \text{otherwise} \end{cases}$
    if $cond_i$ **and** $cond_{i+1}$ **then**
      SWAPVALUES($T(i), T(i+1)$)
      $i := i+1$
    end if
  end for
  return T
end function

function HORIZONTALSPANS (T_{y+1}, T_y, T_{y-1} as arrays of indexes)
  /* Detects horizontal spans for track T_y ****/***
  $H_{spans} := [\,]$ ***/* (output spans) */***
  for $i = 1 \ldots \text{sizeOf}(T_y)$ **do**
    $x_y^a :=$ COLUMNOF($T_y(i)$)
    $x_y^b :=$ COLUMNOF(other route of $T_y(i)$)
    /* Determine whether we can span between x_y^a ***and*** x_y^b ****/***
    $cond_{horz} :=$ **true**
    for $j = (x_y^a + 1) \ldots (x_y^b - 1)$ **do**
      $cond_{sort} := (T_y(j) > T_y(i))$ ***/* sorting condition */***
      /* Test for gap or two-track vertical spans */
      $cond_{vert} := \begin{cases} \textbf{true} & T_y(j) = \emptyset \\ \textbf{true} & T_y(j) = T_{y+1}(j) = T_{y-1}(j) \\ \textbf{false} & \text{otherwise} \end{cases}$
      $cond_{horz} := cond_{horz} \wedge cond_{sort} \wedge cond_{vert}$
      if $cond_{horz} =$ **false then**
        break from for-loop
      end if
    end for
    if $cond_{horz}$ **then**
      Add (x_y^a, x_y^b) to H_{sol}
      $i := i + (x_y^b - x_y^a)$ ***/* Skip span region */***
    end if
  end for
  return H_{sol}
end function

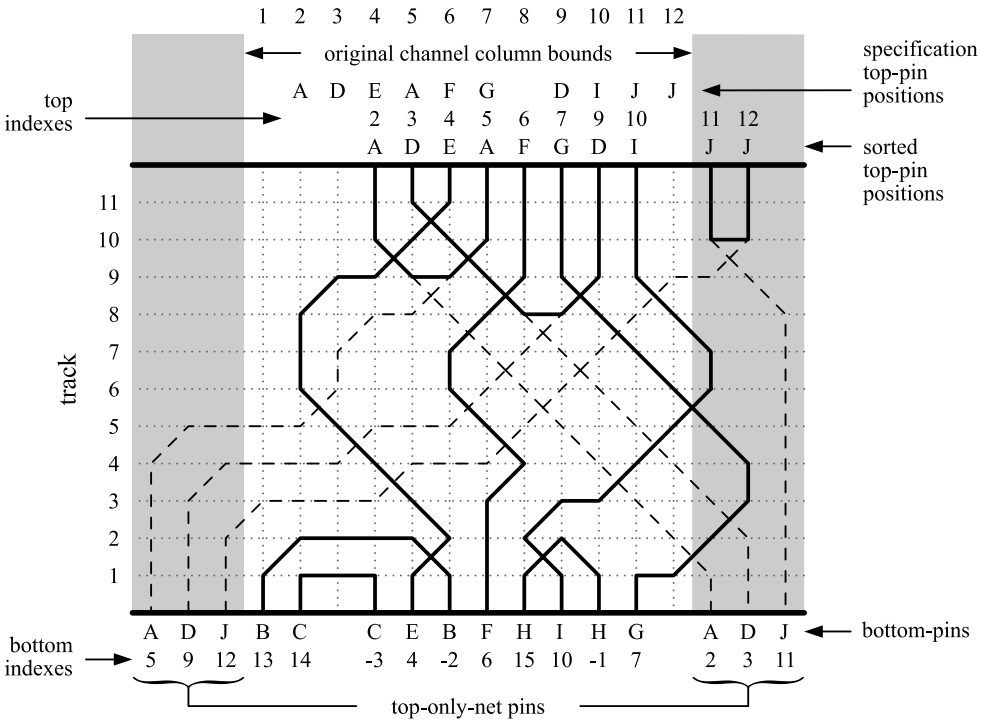


Fig. 6.4 A swap-router channel solution. Shaded region denotes columns outside the initial channel bounds

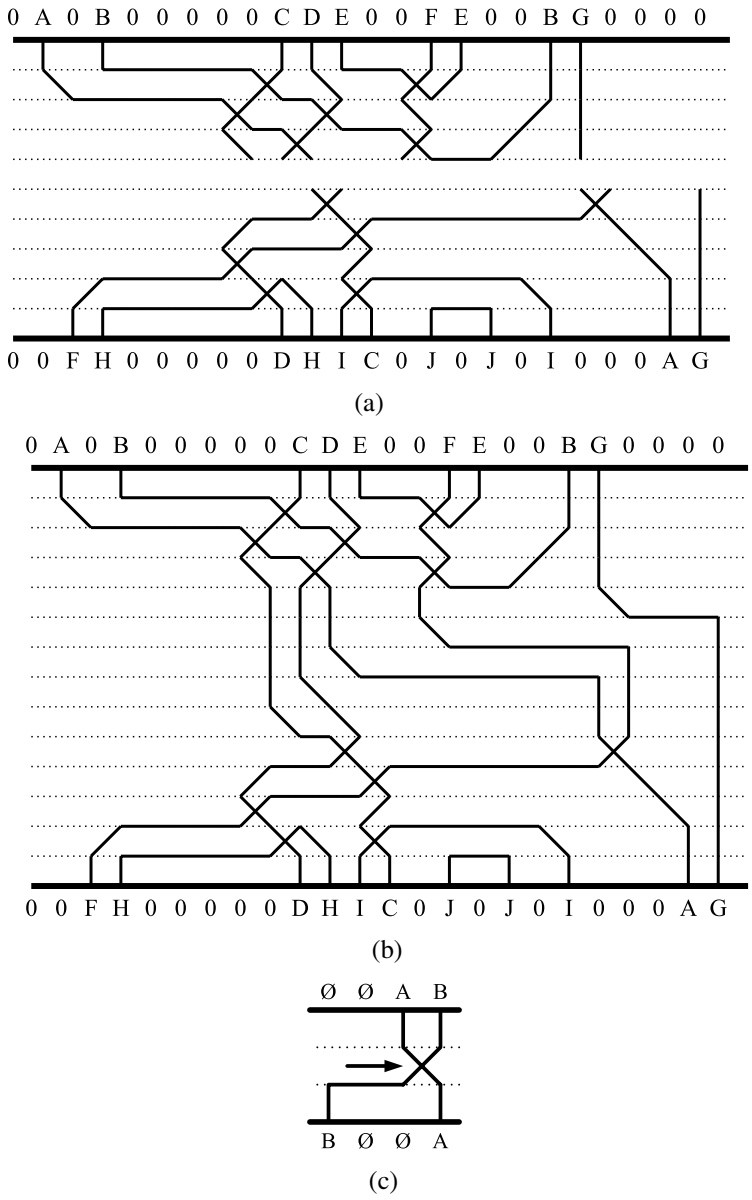


Fig. 6.5 Postsort routing for 2Swap solution: (a) Sorted, but not fully-routed solution (b) After postsort routing (c) Horizontal gap spanning

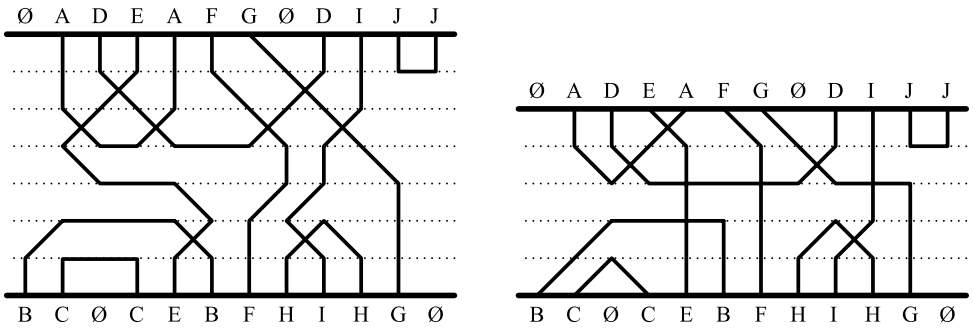


Fig. 6.6 Routing solutions for the same channel instance: (a) 2-sided swap (2Swap) (b) Constrained 2-sided swap (2SwapC)

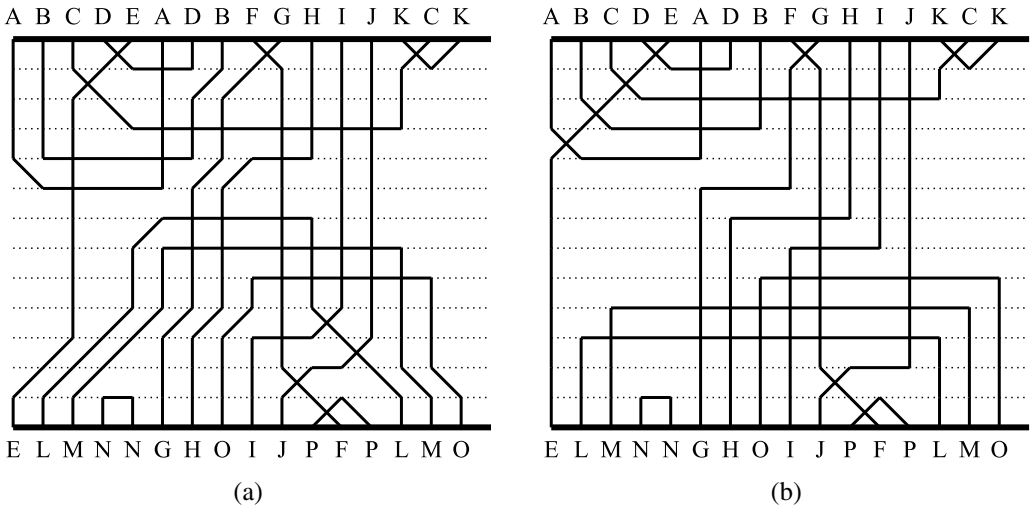


Fig. 6.7 Effect of constraining route movement: (a) 2SwapC allowing convergence towards other pins (12-tracks)
(b) 2SwapC fully-constrained (12-tracks)

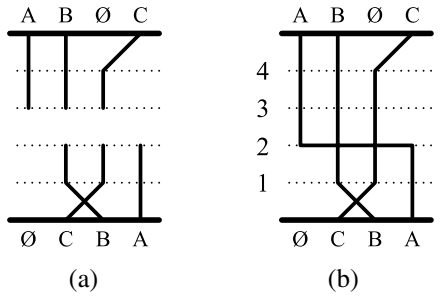


Fig. 6.8 Horizontal connections to complete routing: (a) Net *A* cannot cross *B* or *C* diagonally (b) Horizontal connection across nets

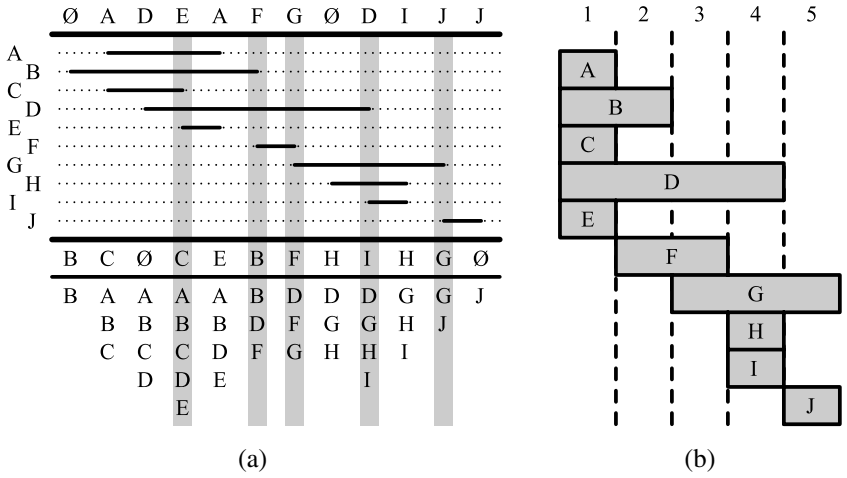


Fig. 6.9 Horizontal constraints and zone representation: (a) Five maximal subsets of signals (b) Resulting five zones

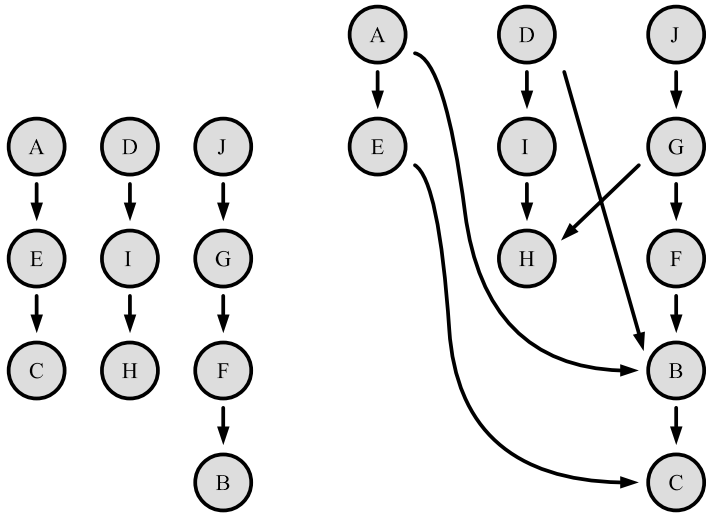


Fig. 6.10 Crossing-constraints modifications to the VCG: (a) Original VCG (b) With crossing constraints

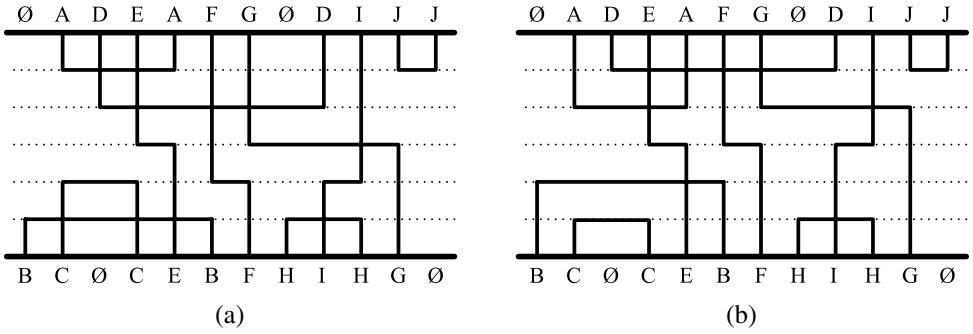


Fig. 6.11 Channel routing solutions under differing constraints:
(a) Track-optimized (5 tracks, 10 crossings)
(b) Crossing-constrained (5 tracks, 8 crossings)

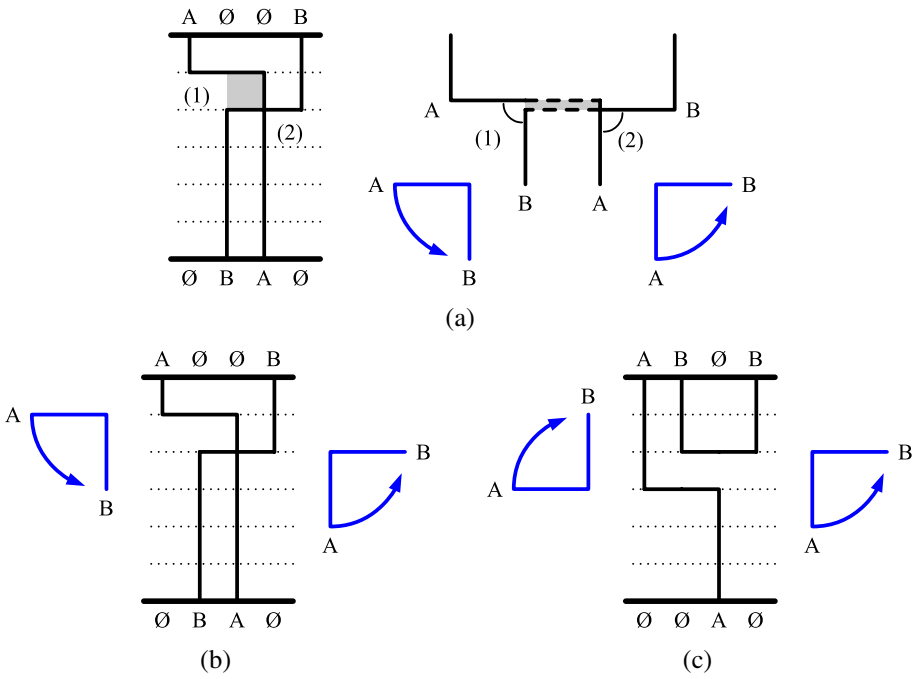


Fig. 6.12 Crossing detection via rotation from *A* to *B*: (a) Rotation direction with respect to pin locations (b) Same rotation direction ⇒ *unavoidable* crossing (c) Opposite rotation directions ⇒ *avoidable* crossing

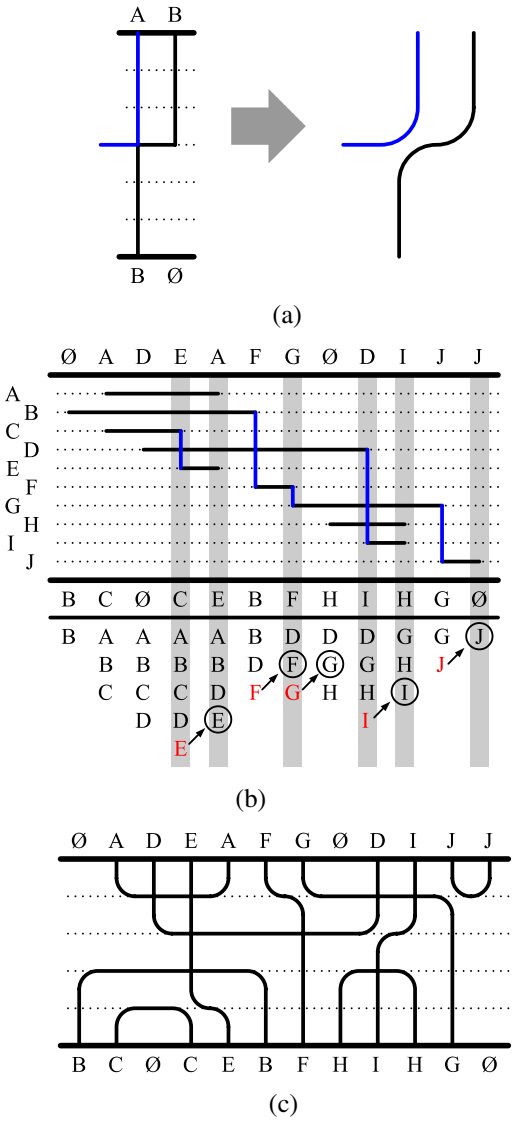


Fig. 6.13 VCGs for Fig. 6.9a and knock-knee extension: (a) Knock-knee implementation (b) Knock-knee-constrained zone representation (c) 4-track routing solution utilizing knock-knees

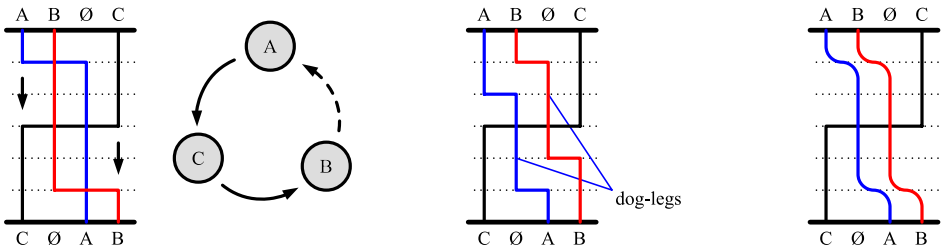


Fig. 6.14 Cycles induced by crossing constraints: (a) Vertical cyclic constraints (b) Dog-leggingavoids crossings (c) Knock-knees avoid additional tracks

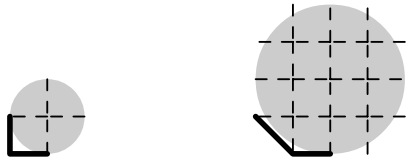


Fig. 6.15 Radius of curvature for grid bends: (a) 90° curve with $r = 1$ loss/bend = 0.368 (b) 135° curve with $r = 2$ loss/bend = 0.135

Table 6.1 Routing benchmark comparison between Yoshimura-Kuh, Constrained 2Swap, and Left-edge + Knock-Knee routers.

Design	Nets	Width	Crossings YK	Crossings (new)	Bend Loss YK	Bend Loss 2SwapC	Bend Loss LE+KK	Tracks YK	Tracks 2SwapC	Tracks LE+KK
alu4.0	104	283	214	128	76.54	**63.20**	76.54	10	12	**10**
alu4.1	150	337	258	140	110.40	**101.11**	110.40	* 11	10	**8**
alu4.2	171	372	663	417	125.86	**116.71**	125.86	18	20	**18**
alu4.3	169	368	555	277	124.38	**117.18**	124.38	12	15	**13**
alu4.4	164	368	703	397	120.70	**118.43**	120.70	15	19	19
alu4.5	167	370	747	441	122.91	**118.06**	122.91	14	20	**17**
alu4.6	186	389	752	434	136.90	**134.90**	136.90	15	16	**15**
alu4.7	144	349	598	424	105.98	**101.63**	105.98	13	17	17
alu4.8	157	354	856	480	115.55	**112.59**	115.55	19	**21**	28
alu4.9	153	348	819	393	112.61	**107.71**	112.61	16	**20**	22
alu4.10	179	368	1082	600	131.74	**129.60**	131.74	22	24	**23**
alu4.11	151	355	404	230	111.14	**105.97**	111.14	10	13	13
alu4.12	154	362	530	252	**113.34**	113.70	**113.34**	11	13	**11**
alu4.13	157	364	520	276	115.55	**112.79**	115.55	11	**18**	20
alu4.14	130	324	243	147	95.68	**87.91**	95.68	8	10	**8**
alu4.15	155	340	373	207	114.08	**103.47**	114.08	11	14	14
alu4.16	146	359	296	170	107.46	**100.31**	107.46	10	12	**11**
alu4.17	151	362	238	126	111.14	**105.07**	111.14	9	11	**9**
alu4.18	140	351	485	259	103.04	**96.01**	103.04	10	13	**11**
alu4.19	126	338	214	128	92.74	**87.50**	92.74	8	11	**9**
alu4.20	143	350	340	184	105.25	**99.69**	105.25	11	16	**15**
alu4.21	141	347	277	145	103.78	**97.36**	103.78	9	11	**10**
alu4.22	133	342	236	108	97.89	**93.14**	97.89	8	9	**8**
alu4.23	157	351	561	335	115.55	**112.10**	117.02	11	15	**13**
alu4.24	147	349	418	194	108.19	**105.40**	108.19	9	12	**11**
alu4.25	140	341	562	316	103.04	**100.99**	103.04	14	17	**16**
alu4.26	156	366	504	240	114.82	**109.55**	114.82	15	17	**15**
alu4.27	141	353	486	228	103.78	**99.49**	103.78	15	17	**16**
alu4.28	91	274	103	59	66.98	**56.36**	66.98	7	8	**7**
alu2.0	93	278	279	157	68.45	**56.46**	68.45	17	18	**17**
alu2.1	142	338	311	159	104.51	**100.99**	104.51	7	10	**8**
alu2.2	178	366	780	440	131.01	**125.18**	131.01	19	24	24
alu2.3	147	355	415	253	108.19	**97.97**	108.19	10	14	**13**
alu2.4	178	388	592	316	131.01	**130.33**	131.01	11	13	**12**
alu2.5	177	381	762	462	130.27	**126.21**	130.27	12	20	**18**
alu2.6	149	352	329	199	109.66	**102.39**	109.66	8	11	**10**
alu2.7	175	375	589	363	128.80	**127.68**	128.80	12	17	**13**
alu2.8	160	362	298	200	117.76	**109.97**	117.76	11	13	**11**
alu2.9	91	273	108	62	66.98	**52.21**	66.98	6	8	**6**
ex5p.0	119	343	244	142	87.58	**73.95**	87.58	10	13	**11**
ex5p.1	145	402	211	129	106.72	**97.60**	106.72	* 11	12	**10**
ex5p.2	207	458	755	399	152.35	**145.64**	152.35	18	24	**23**
ex5p.3	183	427	926	598	134.69	**127.22**	134.69	18	22	**18**
ex5p.4	190	439	912	490	139.84	**136.69**	139.84	18	20	20
ex5p.5	209	457	1362	832	153.82	**145.72**	153.82	18	21	**19**
ex5p.6	183	437	634	342	134.69	**127.31**	134.69	15	16	**15**
ex5p.7	195	437	1061	603	143.52	**138.82**	143.52	18	23	**20**
ex5p.8	175	433	550	320	128.80	**124.76**	130.27	12	19	**13**
ex5p.9	173	425	622	318	127.33	**120.91**	127.33	11	15	**14**
ex5p.10	120	340	231	143	88.32	**72.58**	88.32	9	10	10

N = # nets; Width = channel width

YK = Yoshimura-Kuh; 2SwapC = Constrained 2Swap; LE+KK = Left-edge + Knock Knees

(new) = Either 2SwapC or LE+KK; crossing counts were equal for both routers.

* = indicates YK router performed worse than new routers

CHAPTER 7

THERMAL-AWARE RESYNTHESIS OF PHOTONIC RING RESONATORS

Multicore and network-on-chip architectures comprising multiple disparate subsystems are fast becoming the norm in large system integration. As these systems introduce greater levels of parallelism at the system level, the need for high-speed, high-bandwidth communications becomes a critical factor in overall system operations. Integrated optics will play an important role in such communication fabrics, providing a high-bandwidth, high-fidelity transport layer for inter- and intrachip communications.

A key component of many optical network architectures is wavelength division multiplexing (WDM), enabling multiple channels of data to be transported along the same waveguide. A WDM network operates by assigning (multiplexing) individual channels of data to select wavelengths to be transported via the same data transmission system. Consider a high-level overview of a WDM network depicted in Fig. 7.1a. Channels are assigned to specific wavelengths of light. Wavelength-tuned multiplexing devices—ring resonators in this case—are used to modulate this light, enabling signals (data) to be injected into the waveguide. The light is then routed through the communications fabric. At the endpoints, a demux operation selects for particular wavelengths of light and the signals they carry. This light is then detected at the endpoints by an optical receiver.

At the core of many optical WDM network architectures is the photonic ring resonator. The ring resonator acts as a filter and is used to modulate/multiplex and demultiplex signals on waveguides as depicted in Fig. 7.1a. The device is particularly suited for this application because of the its high Q-factor, providing a high degree of wavelength selectivity. This enables ring-resonator-based WDM systems to support large numbers of channels on the same waveguide.

The filtering response of a ring resonator is periodic with respect to wavelength. Qualitatively, the wavelength range between resonant (filter) peaks is the free spectral range (FSR). The upper limit on the number of channels is determined by 1) the Q-factor of the ring and 2) the free spectral range (FSR) of the ring resonator. In general, the smaller the ring, the greater the FSR; however, losses in ring can contribute to a lower Q-factor, so a balance must be struck. The ring resonators

of a WDM network are usually designed with the same FSR to allow consistent channel spacing without overlap. This channel-wavelength assignment is depicted in Fig. 7.1b.

A major drawback in using ring-resonators for mux/demux operations is their extreme sensitivity to refractive index changes [110]. Process variation, geometric variations, and also thermal effects can shift a ring's resonance off its designed wavelength. Silicon-based ring resonators are especially susceptible to temperature-induced changes to refractive index [111] due to silicon's large thermo-optic coefficient of $dn/dT = 1.86 \times 10^{-4}/^{\circ}K$. In highly integrated systems integrating silicon photonic WDM networks such as those depicted in Fig. 7.2a, heat sources within the chip can cause ring resonators to fall out of resonance. Furthermore, the locality of heat sources means that different rings will be subjected to different temperature conditions such as depicted in Fig. 7.2b. Such external thermal gradients pose significant operational challenges to ring-resonator-based WDM networks.

Contemporary literature proposes active compensation for such refractive index variations (e.g., microheaters [112], carrier-injection based tuning [113] and/or WDM channel remapping [83]); however, these are costly in terms of power and area. Passive-compensation and athermal designs also exist, but require material-level modification or processes [110]. *In this work we present a thermal-aware resynthesis approach for ring-resonator compensation.*

Our approach utilizes a template-based ring-resonator design that enables process-compatible resynthesis to compensate for a predetermined or worst-case temperature gradient. While our approach is not an alternative to active tuning, it complements active tuning. Some amount of active tuning will certainly be required; however, our redesign will ensure lower power/heat for carrier injection or microheater based tuning.

7.1 Ring Resonators

Optical ring resonators are wavelength filtering devices with a notch-filter-type response curve centered around a resonant wavelength. These devices rely on a resonance condition, which causes light within the ring to destructively interfere with light on the coupling waveguides. Alternatively, with two straight waveguides coupled to a single ring, a ring resonator can be used to couple specific wavelengths into or out of a waveguide in a 2x2-switch type operation.

Consider the ring resonator structure depicted in Fig. 7.3a, where a ring of radius r is coupled to a straight waveguide. The coupler is assumed to be symmetrical, and also lossless, implying that the coupling coefficient κ is related to the transmission coefficient t by

$$\kappa^2 + t^2 = 1 \tag{7.1}$$

The ring has an overall length of $L = 2\pi r$, and round-trip loss coefficient α.

A block diagram for the ring resonator structure is depicted in Fig. 7.3b. The electric field amplitude E_{b_2} is the sum of the input electric field E_{a_1} coupled into the ring with coefficient κ, and the round-trip feedback of the ring E_{a_2} coupled back into the ring with coefficient t. Likewise, E_{b_1} is the sum of the t-coupled input signal E_{a_1} and the κ-coupled signal E_{a_2}. These are expressed as

$$E_{b_2} = \frac{-j\kappa E_{a_1}}{1 - t\alpha e^{-j\phi}} \tag{7.2}$$

$$E_{a_2} = E_{b_2}\alpha e^{-j\phi} \tag{7.3}$$

$$E_{b_1} = tE_{a_1} - j\kappa E_{a_2} \tag{7.4}$$

where $\phi = \beta L$ is the phase produced by the round-trip traversal of the ring from b_2 to a_2. The $-j$ factor attached to κ is the result of coupling from one waveguide to another: the latter signal always lags the former by a 90° phase shift.

Combining (7.2)–(7.4) and squaring the result to determine power results in

$$\frac{P_{b_1}}{P_{a_1}} = \left|\frac{E_{b_1}}{E_{a_1}}\right|^2 = \frac{\alpha^2 + |t|^2 - 2\alpha|t|\cos(\phi)}{1 + \alpha^2|t|^2 - 2\alpha|t|\cos(\phi)} \tag{7.5}$$

The value of (7.5) drops to zero (0) when two conditions are met: 1) critical coupling and 2) the resonance condition for the ring.

The first condition, *critical coupling,* occurs when losses in the ring α equal those of the transmission coefficient, i.e., $\alpha = |t|$. Ideally, the ring is considered lossless, i.e., $\alpha \approx 1$, implying that $t = 1$ and from (7.1) that $\kappa = 0$. This produces a paradoxical situation where no energy would couple into the ring in the first place—and could never leave the ring if it did. In practice, however, the curves of the coupler between the ring and straight waveguide are imperfect ($\kappa \neq 0$), enabling the ring resonator to operate as an effective, even if imperfect, filter at the resonant wavelength.

The second condition, denoted the *resonance condition,* constrains the round-trip phase of the ring:

$$\phi = \beta \cdot L = 2\pi \cdot m \tag{7.6}$$

where m is an integer. This causes the term $\cos(\phi) = 1$ and in conjunction with critical coupling $P_{b_1}/P_{a_1} = 0$. The dependence of ϕ on β implies that resonance is wavelength and waveguide (i.e., effective index) dependent. A useful relationship is derived:

$$\beta \cdot L = \frac{2\pi n_{eff}}{\lambda_0} \cdot L = 2\pi \cdot m \qquad \Rightarrow \qquad n_{eff} \cdot L = m \cdot \lambda_0 \tag{7.7}$$

7.1.1 Free Spectral Range

As noted earlier, ring resonators have a free spectral range (FSR), the $\Delta\lambda$ between resonance peaks depicted in Fig. 7.1b. More formally, the FSR is defined as

$$\text{FSR} = \frac{\lambda_0^2}{n_g L} \tag{7.8}$$

where n_g is the dispersion-dependent *group index* of the waveguide

$$n_g = n_{eff} - \lambda \frac{\partial n_{eff}}{\partial \lambda} \tag{7.9}$$

7.2 Thermal Compensation for Ring Resonators

Consider the reference ring resonator structure depicted in Fig. 7.4. This ring resonator is a 4-port structure, enabling the ring to demultiplex light entering from the input-port to the *drop*-port at the same resonant wavelength(s) as the 2-port ring; nonresonant wavelengths pass to the *through*-port. We define $n_{eff,0}$ as the effective index of the waveguide in the *absence* of a thermal variation, i.e., $T = T_0$. A change in temperature ΔT results in a change to the effective index due to the thermo-optic properties of the waveguide materials, notably in the waveguide's silicon guiding layer. This change in refractive index, in turn, causes a change to the waveguide's propagation constant. Let $n_{eff,\Delta T}$ and $\beta_{\Delta T}$, respectively, denote the effective index and propagation constant in the presence of a temperature change $T = T_0 + \Delta T$:

$$n_{eff,0}\Big|_{\Delta T} = n_{eff,\Delta T} \qquad\qquad \beta_0\Big|_{\Delta T} = \beta_{\Delta T} \tag{7.10}$$

The change in propagation constant causes the ring to shift out of resonance as governed by (7.6). This shift can be compensated in multiple ways: 1) active compensation using external effects, such as heat, electric fields, etc.; 2) material-level compensation; and 3) geometric changes to the device structure or waveguide.

7.2.1 Active Compensation

Active compensation (tuning) utilizes external effects to change the optical properties of materials. For SOI ring resonators, active tuning is usually implemented via microheaters [112]—exploiting silicon's relatively large thermo-optic coefficient of $1.86 \times 10^4/^\circ K$. Tuning can also be performed by using carrier injection by applying a DC-bias to the ring [113]. In all such active compensation methods, outside energy and feedback are required to prevent resonance wavelength drift and offset.

Channel-remapping approaches [83] have been proposed as a means to reduce active power. These approaches reassign WDM channels to different rings depending on their perturbed resonance

conditions and require active tuning power. The drawback to this approach is that a larger number of rings are necessary to enable channel remapping, and rings may conflict should their filtering response be similar.

7.2.2 Material-Level Passive Compensation

Permanent compensation can be achieved by manipulating the optical properties of a waveguide's materials—i.e., "trimming." Trimming is often performed by affecting the waveguide's cladding layer through stress or additional material layers [114], [115] or by introducing materials that counteract the thermo-optic coefficient of silicon, such as polymers on narrowed waveguides [116]. Such methods require additional materials and lithographic processes that increase the cost and complexity of fabrication. In addition, materials such as polymers on narrowed waveguides can affect mode confinement, disallowing sharp bends [110].

7.2.3 Geometric Compensation

Our emphasis is on manipulations to the geometry of the device or waveguide structure—*geometric compensation.* This is an attractive option because modifications to devices can be performed without special process steps, materials, or relying on active compensation. For ring resonators, this involves changes to parameters such as the ring length, or the profile-dimensions of the ring's waveguide.

7.2.3.1 Compensation using ring length

Consider the racetrack (ring) resonator depicted in Fig. 7.5. The racetrack structure is designed to be equivalent to that of Fig. 7.4 in terms of its resonant wavelength λ and total ring length L. The structure, however, incorporates two compensation regions of length L_{comp} such that $L = 2(\pi r_{comp} + L_{comp})$ satisfies (7.6). By varying the parameter L_{comp}, facilitating a ring-length change, we can compensate for changes in refractive index.

Incorporating the above L into (7.7) we have

$$m\lambda_0 = 2(\pi r_{comp} + L_{comp})\left(n_{eff,0} + \frac{dn}{dT}\Delta T\right) \tag{7.11}$$

where dn/dT is the thermo-optic coefficient of the waveguide structure, which is on order of that of the guiding silicon layer ($\frac{dn}{dT} = 1.86 \times 10^{-4}/^{\circ}K$). Rearranging for L_{comp} gives us

$$L_{comp} = \frac{m\lambda_0}{2\left(n_{eff,0} + \frac{dn}{dT}\Delta T\right)} - \pi r_{comp} \tag{7.12}$$

From (7.12) we wish to determine the change in L_{comp} as a function of temperature change. This leads to the expression

$$\Delta L_{comp} = \frac{m\lambda_0}{2\left(n_{eff,0} + \frac{dn}{dT}\Delta T\right)} - \pi r_{comp} - \left(\frac{m\lambda_0}{2n_{eff,0}} - \pi r_{comp}\right) \tag{7.13}$$

$$= \frac{m\lambda_0}{2}\left(\frac{1}{n_{eff,0} + \frac{dn}{dT}\Delta T} - \frac{1}{n_{eff,0}}\right) \tag{7.14}$$

Equation (7.14) implies that while values of ΔT produce small differential quantities, and likewise small changes to ΔL_{comp}, these variations can be multiplied in effect by increasing the ring length through parameter m. For example, consider a ring resonator constructed using a waveguide with dimensions 400 nm $\times$ 180 nm, with $n_{eff,0} = 1.9065$ at the resonant wavelength $\lambda_0 = 1550$ nm. If this waveguide undergoes a change in temperature $\Delta T = 10^{\circ}K$ we have

$$\Delta L = \frac{m \cdot 1550 \text{ nm}}{2} \cdot \left(\frac{1}{1.9065 + 10^{\circ}K \cdot 1.86 \times 10^{-4}/^{\circ}K} - \frac{1}{1.9065}\right) \tag{7.15}$$

$$= m \cdot (-0.39620 \text{ nm}) \tag{7.16}$$

The effect of the temperature change is therefore so small that ring length must amplify its effects in order for compensation to be effective. If we assume the lithographic process can fabricate features on order of ≈ 10 nm, the minimum value of m is ≈ 25, implying $L \approx 20.3$ μm, for compensation at relatively coarse $10^{\circ}K$ increments.

Modifying the ring length does have drawbacks. Ensuring precision requires that L must have a minimum length, possibly impacting ring specifications such as FSR, which is inversely proportional to L. Also, in changing the length of the ring, the dimensions of the entire ring structure must be changed. This in turn does not lend itself to a template-based methodology.

7.2.4 Compensation Using Waveguide Width

The guiding properties of a waveguide are dependent on both materials and geometry of the waveguide profile. As light propagates down a waveguide, variations to this dielectric profile, as a function of length, affects the propagation parameters of guided light within the structure. By introducing *perturbations* to the waveguide structure, a designer can exploit this mechanism to control the transmission and reflection properties of the waveguide. Purposely introduced waveguide perturbations are usually used in the context of creating resonant structures of periodic dielectric variations. For example, structures such as Bragg gratings [117] control for the width and spacing of these periodic perturbations to provide wavelength filtering.

We control for the waveguide *width* for a fixed compensation length (denoted a "notch") in order to introduce a perturbation that enables ring-resonator tuning. This is depicted in a 3D representation

of a waveguide in Fig. 7.6b. The novelty of this approach is that we can control and tune for round-trip phase within a ring by changing the propagation properties—via the waveguide width—in a short *subsection* of the ring. Also, by controlling for only this subsection of the overall ring length, we have greater control over the phase tuning. This is important for ensuring that waveguide width tuning remains feasible for semiconductor process resolutions. Finally, for rings of a given resonant wavelength, we construct a single template ring. Tuning is performed only by modifying the width of the compensation region without requiring additional resynthesis procedures over the other segments of the ring.

We propose a ring (racetrack) resonator template to enable thermal compensation using 2D lithographic methods. Consider the racetrack (ring) structure depicted in Fig. 7.6a. The racetrack waveguide contains a single section of length L_{comp} as its compensation region. This section of waveguide is varied in width to offset temperature-induced changes to refractive index. In effect, we counteract material changes with geometric changes. The resonator template is bound by three conditions:

- **Resonance Condition:** The overall structure and effective index must satisfy the resonance condition (7.6).
- **Compensation Region Length:** The notch must minimize reflections to avoid signal loss.
- **Process Manufacturability:** Variations to the width must be feasible in current processes.

7.2.4.1 Resonance condition

Satisfying the resonance condition for the ring is implicit as this ensures that the ring is tuned for a specific wavelength. As the ring is broken down into sections, the phase term for the ring is the integral of the refractive index changes around the ring:

$$\phi = 2\pi m = \int_0^L \beta(z)dz \tag{7.17}$$

$$= \beta_{\Delta T}\left(2\pi r_{comp} + L_{comp}\right) + \beta_{comp} L_{comp} \tag{7.18}$$

where $\beta_{comp} = 2\pi n_{eff,comp}/\lambda_0$ in the compensation waveguide length L_{comp}.

7.2.4.2 Notch effects and reflection minimization

The waveguides in our devices, such as Fig. 7.6b, are 3D in nature, but do not vary in height. Viewed from the top-down, we consider device structures such as ring resonators as 2D waveguide structures waveguide composed of three material regions (“layers”) as depicted in Fig. 7.6b. We denote the guiding structure layer as n_f, and the substrate layers n_s, where $n_f > n_s$. By varying the width of a waveguide, the effective index of the waveguide profile is altered.

Consider the structure depicted in Fig. 7.7, an "unrolled" representation of the ring structure depicted in Fig. 7.6a and a 2D representation of Fig. 7.6b. In place of a single effective index term for a region of the waveguide, the waveguides of the ring are now described in terms of their constituent materials' n_f and n_s refractive indexes and width and length parameters. We denote the compensation waveguide region as a dielectric "notch" of width w_{comp} and length L_{comp}. This notch is defined as the region of width w_{comp} extracted from the unperturbed waveguide. Therefore, the width of the notch-region *waveguide* is $w - w_{comp}$; the width of the noncompensated regions is w. In terms of the z-dimension, the notch begins at $z = -a$ and ends at $z = a$ for a total length $L_{comp} = 2a$.

The notch changes the properties of the original $n_{eff,\Delta T}$ waveguide, representing a *perturbation* on the waveguide structure. The effect of this perturbation is such that when forward-moving, guided light strikes the notched region, energy is coupled into different modes. These modes can be a combination of 1) the same mode, but with changes to properties such as phase, 2) intermode coupling, if the waveguide supports additional modes, or 3) a combination of (1) and (2) *reflected* into modes of the *backwards* wave.

We analyze Fig. 7.7 under the assumption that guided modes are TE in nature and that our waveguide is *single-mode.* This simplifies our analysis to consider only coupling to the forward and backwards traveling waves of the waveguide, A^+ and A^-, respectively. Further analysis of perturbations on multimode waveguides can be found in [38].

The notch causes a *polarization perturbation* in the waveguide: the product of the change in dielectric constant and the electric field of the forward wave. This perturbation only occurs over the notched region, and therefore

$$P_{pert} = \begin{cases} \Delta\varepsilon E_y & \text{within notch} \\ 0 & \text{otherwise} \end{cases} \tag{7.19}$$

$$\Delta\varepsilon = \varepsilon_0 \left(n_s^2 - n_f^2\right) \tag{7.20}$$

This perturbation causes coupling between modes of the waveguide. In this single-mode waveguide, coupling will occur to the same mode of the forward-traveling wave and/or reflect into the same mode of the backwards traveling wave. For the forward wave, we have

$$E(z) = \frac{A^+(-a)}{2} e^{-j(\beta+\kappa)z} \qquad \text{for } -a < z < a \tag{7.21}$$

where κ is a coupling constant produced by the perturbation. In effect, the perturbation effects a *phase change* in the resulting wave—the effect we exploit for the purposes of compensation.

The notch also causes a reflection into the backward wave A^-. The amplitude of this wave, derived in [38], is dependent on the length a:

$$A^-(-a) = \frac{-j\kappa A^+}{\beta} \sin(2\beta a) \tag{7.22}$$

We wish to minimize reflections, and we note that (7.22) is periodically minimized as a function of a. Therefore, when setting $A^-(-a) = 0$ we have

$$2\beta a = q\pi \tag{7.23}$$

$$a = \frac{q\pi}{2\beta} \tag{7.24}$$

where q is an integer. Given $\beta = 2\pi n_{eff}/\lambda_0$ and $\lambda = \lambda_0/n_{eff}$, (7.24) implies that the length $a = q\lambda/4$. Recall that the notch length $L_{comp} = 2a$. Our reflection-minimizing notch-length is therefore

$$L_{comp} = \frac{q\pi}{\beta} = \frac{q\lambda}{2} \tag{7.25}$$

The radius of the curved regions of the template, r_{comp}, is derived by substituting (7.25) into (7.18) and assuming that $\beta = \beta_{\Delta T} \approx \beta_{comp}$

$$\phi = 2\pi m = \beta\left(2\pi r_{comp} + \frac{q\pi}{\beta}\right) + \beta\frac{q\pi}{\beta} \tag{7.26}$$

$$= 2\pi\left(\beta r_{comp} + q\right) \tag{7.27}$$

$$r_{comp} = \frac{m-q}{\beta} \tag{7.28}$$

7.2.4.3 Process manufacturability

Lithographic processes have resolution limitations that prevent the fabrication of exact widths for notches. We therefore perform additional design space exploration for compensation with respect to process manufacturability. Let $w_{process}$ be the minimum unit width that may be fabricated by the lithographic process. The unit $\Delta\lambda$ shift for the compensation-enabled ring will therefore be defined by

$$w_{comp}^{unit} = \text{floor}\left(\frac{\Delta w_{comp}}{w_{process}}\right) \cdot w_{process} \tag{7.29}$$

Though reflections are minimized with respect to notch length, a notch will still cause reflections if it is too deep. Therefore, in order to enable compensation over wide temperature ranges, the compensation region must be lengthened by a multiple of the minimum unit compensation length $\lambda/2$. A benefit of longer compensation lengths is that each $\lambda/2$ subsection of the compensation lengths may be varied independently to effect compensation in a more precise manner. In addition, a longer compensation region reduces the need for deeper notches, which can affect loss. Also, while the ring depicted in Fig. 7.6a only uses a single compensation region, the template ring already includes two possible compensation regions.

7.2.5 Methodology and Demonstration for a WDM Ring Resonator

We demonstrate how thermal compensation is achieved by designing a template ring resonator device for a WDM network. This particular resonator structure is designed to filter $\lambda_0 = 1550$ nm. Each channel of the WDM network utilizes a ring with a specific resonance wavelength λ_0, with sufficient FSR for multiple channels. In this example, we choose an FSR of approximately 15 nm. The system utilizes an SOI ridge waveguide, 400 nm wide and 180 nm in height. For the waveguide profile, we measure $n_g \approx 4.63$ using a mode solver [118]. This leads to a desired ring length

$$L_{FSR} = \frac{(1550 \text{ nm})^2}{4.63 \cdot 15 \text{ nm}} \tag{7.30}$$

$$\approx 34.6 \ \mu\text{m} \tag{7.31}$$

Using a mode-solver, we measure an effective index and β for the waveguide profile to be

$$n_{eff} = 1.9065 \tag{7.32}$$

$$\beta = \frac{2\pi n_{eff}}{\lambda_0} = 7728317.9/\text{ m} \tag{7.33}$$

From the value of β we derive a ring length $L \approx L_{FSR}$ that satisfies the resonance condition (7.6):

$$m = \text{floor}\left(\frac{\beta L_{FSR}}{2\pi}\right) = 42 \qquad \Rightarrow \qquad L = \frac{42 \cdot 2\pi}{\beta} = 34.15 \ \mu\text{m} \tag{7.34}$$

The total length L enables us to construct a ring resonator structure incorporating a compensation region. We choose a compensation length $L_{comp} = 3\lambda/2$, three (3) times the minimum notch length ($\lambda/2$), in order to magnify the effects of the perturbation produced by the notch narrowing.

We construct and simulate a 3D material representation of our designed ring resonator using an FDTD-type simulator such as Lumerical MODE Solutions or FDTD Solutions. Light is injected into the input of the structure over a range of wavelengths, and the transmission response is measured at the through- and drop-ports of the structure. In Fig. 7.8, we plot the filtering response (measured at the through-port) of the *reference* ring, where $\Delta T = 0, w_{comp} = 0$ nm, reflecting the baseline resonant wavelength for this structure.

On the same plot, we also show the wavelength response as a function of 1) different temperatures ($T = 300^\circ K + \Delta T$ for $\Delta T = 0, 20, 40, 60^\circ K$) and fixed $w_{comp} = 0$ nm and 2) *width-narrowing* of the waveguide in the compensation region by a particular amount (w_{comp}), for fixed $\Delta T = 0^\circ K$. Observe how temperature increases cause a *positive* resonant wavelength shift. Conversely, narrowing the width of the compensation region's waveguide—by increasing w_{comp}—results in a *negative* wavelength shift. *We can therefore compensate for a static temperature change by narrowing the waveguide of the compensation region.*

The template-ring is compensated for *arbitrary* temperature changes by deriving two sets of sampled values—ΔT vs. $\Delta\lambda$, and w_{comp} vs. $\Delta\lambda$—and coupling these two to derive w_{comp} from ΔT.

The two tables are derived independently. For ΔT vs. $\Delta\lambda$, we fix $w_{comp} = 0$ nm and simulate the ring under different ΔT. Over the device structure, the ΔT is multiplied by each material's thermo-optic coefficient, changing the refractive indexes of the waveguide structures and effecting wavelength shifts that we sample at the output ports over the range input wavelengths.

Likewise, we simulate the ring under different values of notch-narrowing w_{comp} while fixing $\Delta T = 0$, to determine w_{comp} vs. $\Delta\lambda$. For the example ring, these tables are presented in Table 7.1.

From the tables, w_{comp} is derived from an arbitrary ΔT by: 1) interpolating the wavelength shift $\Delta\lambda_{\Delta T}$ using Table 7.1(a) and 2) using $\Delta\lambda_{\Delta T}$ to interpolate a value of w_{comp} from Table 7.1(b). With even a sparse number of sampled points, meaningful thermal compensation can be performed.

For example, if we assume a temperature differential of $\Delta T = 27^{\circ}K$, the predicted interpolated $\Delta\lambda_{\Delta T}$ is calculated to be

$$\Delta\lambda_{\Delta T} = 2.1080 \text{ nm} + (3.0132 \text{ nm} - 2.1080 \text{ nm}) \frac{(27-20)^{\circ}K}{(30-20)^{\circ}K} = 2.7417 \text{ nm} \tag{7.35}$$

We compensate for this positive wavelength shift with an equal negative wavelength shift by interpolating w_{comp} from Table 7.1(b):

$$w_{comp} = 40 \text{ nm} + (50-40) \text{ nm} \frac{(2.7417 + 2.5021) \text{ nm}}{(-3.2012 + 2.5021) \text{ nm}} = 43.4265 \text{ nm} \tag{7.36}$$

This yields $w_{comp} \approx 40$ nm. The wavelength response comparing of the compensated system with the reference system is depicted in Fig. 7.9. By inspection, the two response curves are in very close agreement.

This example and supporting methodology demonstrate that we can compensate for temperature-induced resonant-wavelength shifts in ring-resonators by varying the width of waveguides in subsections of the ring. The benefit of this approach is that a single template can be constructed that accommodates a wide range of temperature variations, while remaining relatively precise. The example also only uses one of the possible compensation regions, and the precision of compensation could be further improved by varying subsections of each compensation length.

7.3 Conclusion

We have presented a methodology for constructing a compensation-enabled ring resonator template. Process-compatible modifications to the waveguide structure enable this template structure to counteract the effects of temperature changes across the ring. This template requires only minimal changes to effect meaningful temperature compensation, and can be applied to a wide range of temperature offsets with useful precision. Such thermal-aware resynthesis reduces the amount of active compensation required to tune ring resonators. This, in turn, will save power required for integration into opto-electronic hybrid systems.

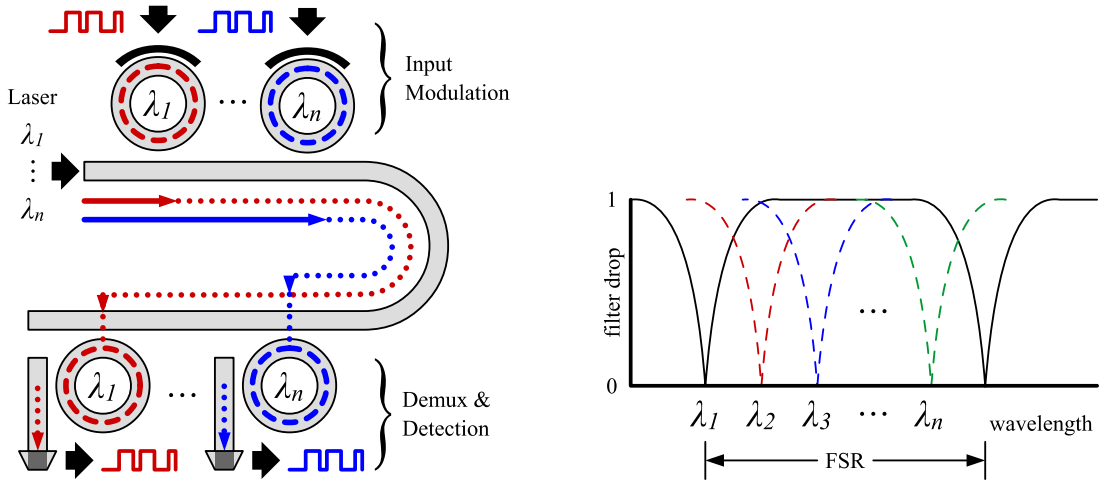


Fig. 7.1 WDM network overview and channel assignments: (a) Modulation and demultiplexing/detection (b) Channel wavelengths within the FSR of a WDM ring resonator

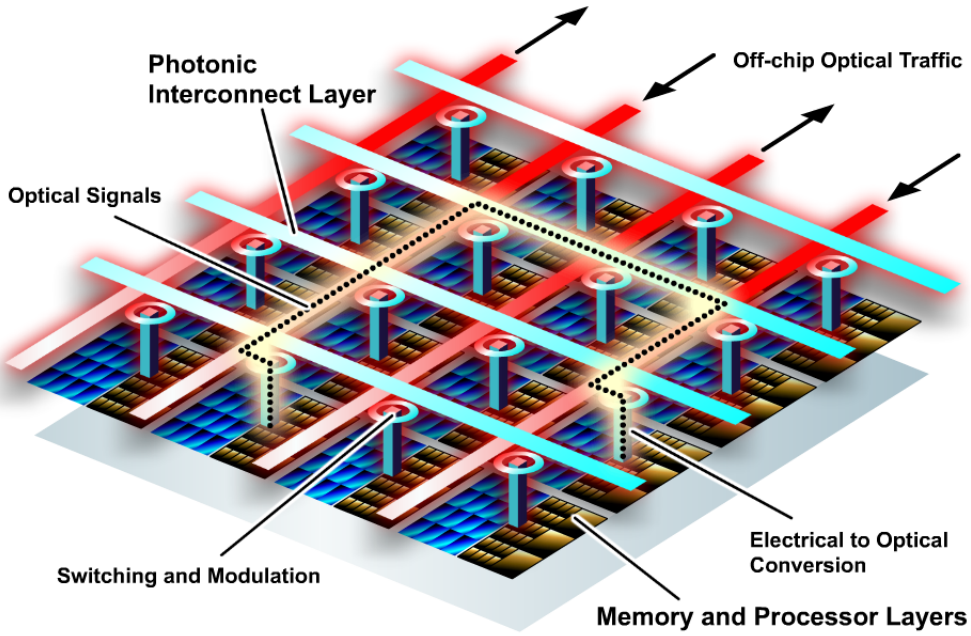
Table 7.1 Wavelength shifts due to ΔT and w_{comp} for example ring.

ΔT	$\Delta\lambda$
0 ° K	0 nm
10 ° K	1.1035 nm
20 ° K	2.1080 nm
30 ° K	3.0132 nm
40 ° K	4.0202 nm
50 ° K	5.0285 nm
60 ° K	5.9371 nm

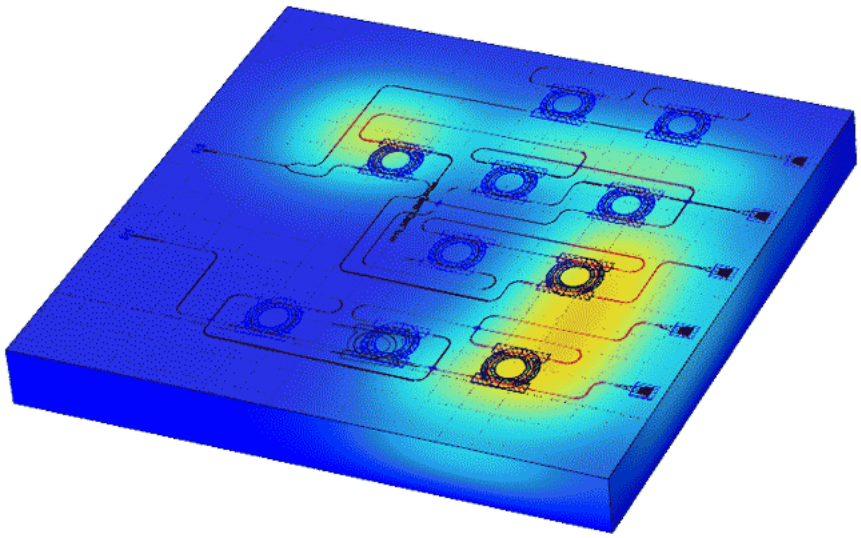
(a) ΔT *vs* $\Delta\lambda$

w_{comp}	$\Delta\lambda$
0 nm	0 nm
10 nm	-0.5011 nm
20 nm	-1.1019 nm
30 nm	-1.8023 nm
40 nm	-2.5021 nm
50 nm	-3.2012 nm
60 nm	-3.9995 nm

(b) w_{comp} *vs* $\Delta\lambda$



(a)



(b)

Fig. 7.2 Integration of ring-resonator networks: (a) Photonic interconnect layer in an integrated chip design (b) On-chip heat sources creating thermal gradients across an optical substrate

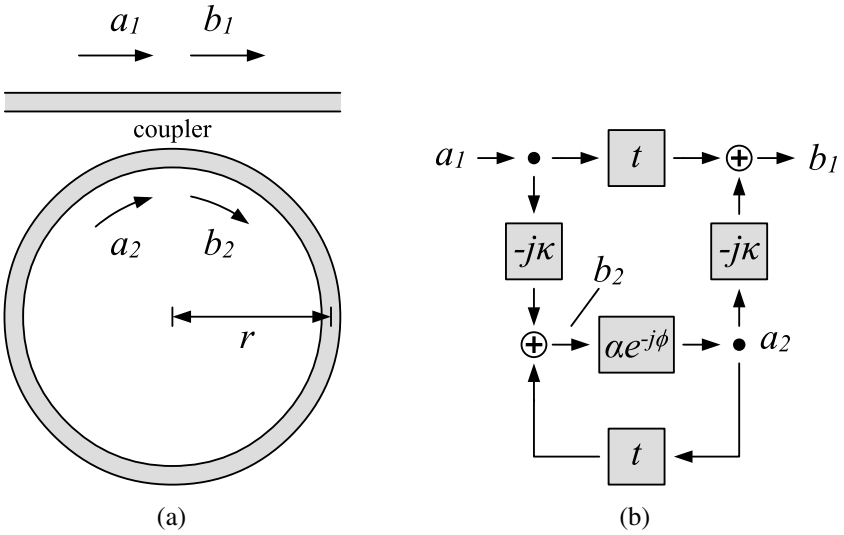


Fig. 7.3 Structure of an Optical Ring Resonator: (a) Ring resonator (b) Block diagram

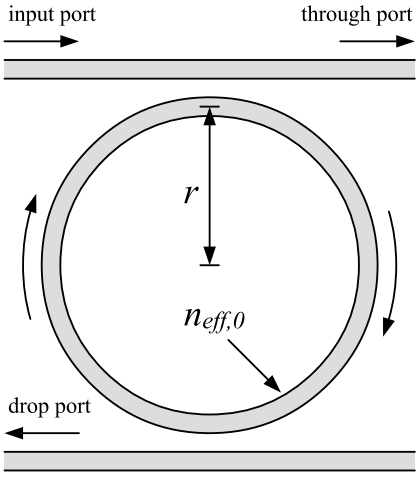


Fig. 7.4 Original uncompensated ring

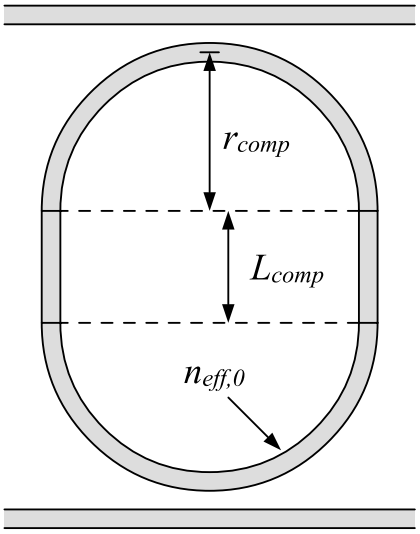


Fig. 7.5 Racetrack (ring) resonator

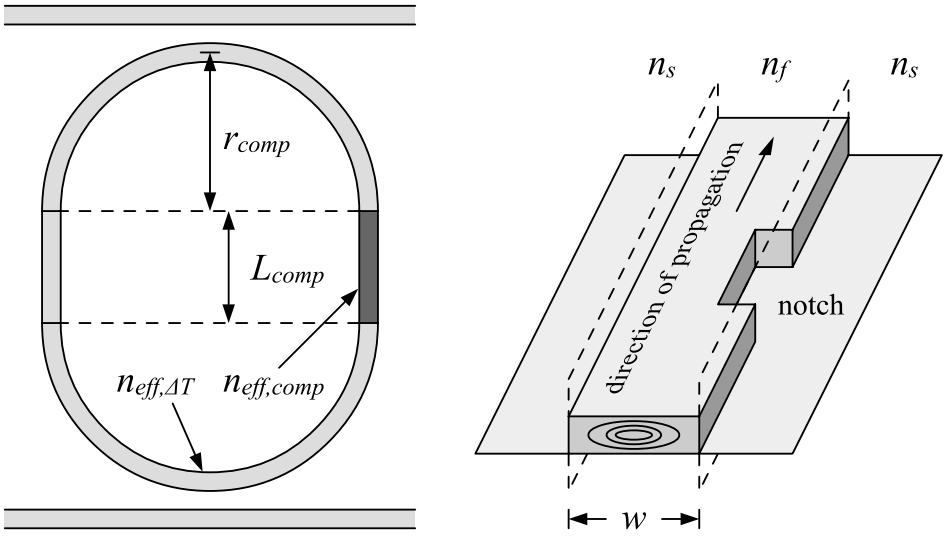


Fig. 7.6 Racetrack (ring) resonator with compensation region: (a) Racetrack (ring) resonator with compensation region (b) 3D representation of a waveguide with notch; 2D regions of material indexes

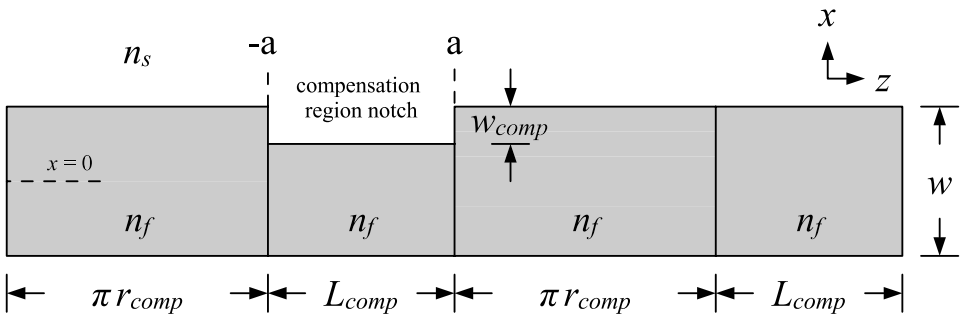


Fig. 7.7 An unrolled view of a ring resonator

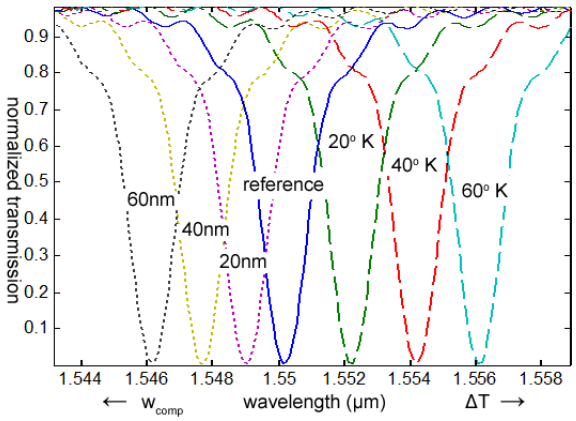


Fig. 7.8 Wavelength shifts due to changes in ΔT and w_{comp}

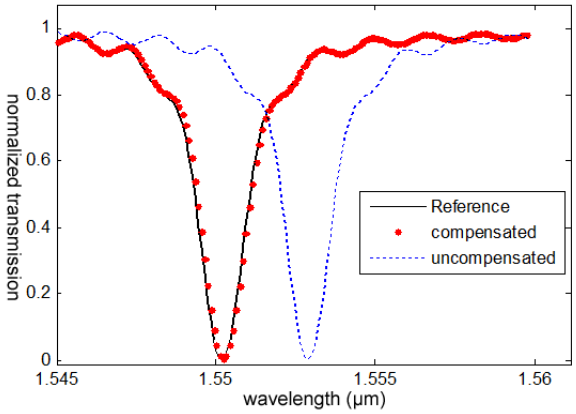


Fig. 7.9 Compensation for $\Delta T = 27°K$ with $w_{comp} = 40$ nm vs. unaffected reference ring

CHAPTER 8

CONCLUSION

This dissertation presents design automation for integrated optic system design. We demonstrate that design automation, in the manner of VLSI, is possible through a building-block methodology requiring technology-specific constraints and objectives. This building-block methodology is incorporated into an EDA-inspired, integrated optics design flow, which is broken down into behavioral and physical synthesis stages. In each of these stages we provide techniques and methodologies enabling automated synthesis of optical networks.

In behavioral synthesis, we model optical switching devices as logical building-blocks in the form of crossbar gates. For these gates, a Boolean logic function composition methodology is presented—"virtual gates"— enabling the synthesis of arbitrary factored Boolean expressions. For our optical device model, multi-output expression sharing is not possible using conventional factorization-based synthesis. We therefore invent a new XOR-based decomposition technique, enabling common subexpression extraction. This expression sharing minimizes devices counts, reducing signal loss.

Subsequent to behavioral synthesis, optical devices must be placed and interconnected using global and detailed routing. We utilize a row-based placer in a standard-cell methodology for integrated optic device placement. This placement gives rise to routing channels. Routing is performed in two stages where a global routing solution is proposed to derive a net-routing topology that minimizes crossings, bends, and congestion. This gives rise to channel definitions that are then solved using channel routing techniques for integrated optics.

We present two techniques for signal-loss-constrained channel routing for integrated optics. Our first channel router is based on sorting, whereas another is based on constrained left-edge style routing. Experimental results demonstrate that a sorting based router produces loss-optimized solutions at a slight expense of area. On the other hand, our constrained left-edge style channel router, incorporating technology-specific extensions, produces area-optimized solutions. Therefore, the routers can be leveraged based on the optimization desired.

Integrated optics will also be leveraged in hybrid opto-electronic systems, especially for optical communication networks. Electronic switching may create temperature-hotspots that will interact with the optical substrate, producing a thermal gradient. Optical devices such as ring resonators, used in many optical network architectures, are extremely sensitive to temperature-induced changes in refractive index. This causes their resonant wavelength to shift out of spec, affecting their designed operation. To overcome this problem, we present a thermal-aware, physical resynthesis template that exploits perturbation theory applied to waveguide dimension changes. We provide an automated approach that analyzes simulated data over an uncompensated ring and derive redesign parameters to enable and achieve thermal compensation.

Overall, using this design flow, we have demonstrated that optical design can be taken from the behavioral level to the physical level within an automated framework. We have also presented resynthesis techniques enabling integration into hybrid systems. Our techniques are applied to our own designs as well as conventional VLSI benchmarks, and we have fabricated a silicon photonic design implementing our own optical logic. Design automation for integrated optics is therefore not only feasible, but a necessary step in the development of the technology and its applications.

8.1 Future Work

This work has only touched upon a small portion of what is possible in design automation for integrated optics. Tremendous opportunities abound in future research in this area, enabling integrated optics to reach its full potential as a communications and computation technology.

8.1.1 Thermal-Aware Placement and Analysis

While this dissertation demonstrates the use of EDA-style methodologies and makes contributions at the logic and routing levels, optical device placement is a problem that requires significant amounts of further research. More importantly, thermal constraints must be incorporated into the placement models. There has been substantial research in the VLSI domain on thermal-aware placement; however, the integrated optics problem is different. The thermal-placement problem for VLSI optimizes a layout to evenly spread out the heat sources as a means to mitigate temperature hotspots. In integrated optics, however, heat sources are generally external, affecting the optics layers and creating integration challenges. Though we have proposed solutions for thermal compensation of preplaced devices, external thermal constraints must be analyzed and incorporated into the placement models themselves.

The first step in an integrated placement approach is to derive the temperature gradient as it affect the optical layer. We can assume a 3D die-stacking scenario whereby a mixed or multicore system is integrated with a photonic communications or routing layer. A thermal analysis will be

performed based on the workload and circuit-activity characteristics of the system cores, producing temperature hotspot information. The temperature gradient itself is a function of both heat sources and sinks and the materials between them. In our model of the ring resonator, we assumed a single temperature across the entire ring. However, additional analysis can be performed to characterize optical devices in terms their constituent material thermal properties, especially for nonuniform gradients. For example, in SOI optical devices, cladding materials are constructed of thermal insulators as compared, whereas the guiding layers are highly thermally conductive. The movement of heat through such a device may affect how it may be placed.

With such thermal characterization in place, we can utilize this information to drive optimization techniques. Our ring resonator resynthesis approach demonstrates that resynthesis is a viable method for compensating for the effects of a static temperature change. However, in placement we have the flexibility of choosing or avoiding certain temperature conditions. We can speculate on the optimization criteria such thermal-aware placement techniques may employ.

At the most basic level, seeking out areas of low temperature may be desirable as a means to reduce the effects of high temperatures on optical devices, e.g., thermal expansion, losses, or outside power needed for compensation. However, given the ability to resynthesize devices, it may be more useful to seek out areas of consistent temperature values or even areas with *stable* temperatures—reducing the temperature range over which active tuning must be performed. Overall, with sufficient data, a combination of thermal interactions can be utilized provided the underlying optimization techniques support them.

Thermal gradients and optimization criteria must then be coupled into a placement framework. Here we may borrow from conventional EDA placement techniques while incorporating thermal constraints and optimization goals into the model. In a force-directed placement approach such as [119], the thermal gradient map and/or hot spots serve as attractive or repulsive forces within the framework. This will be in addition to forces that ensure locality to endpoints (e.g., a filter near its receiver device) and forces that prevent overlap and obstacle avoidance. Thermal constraints can also be incorporated into a coordinated place and route type framework [120] by utilizing a thermal gradient map in place of congestion maps for placing and routing optical GCells. Finally, thermal placement is amenable to simulated annealing based placement approaches. In effect, the idea is to use the 2D thermal gradient itself as a cost in the annealing schedule to optimally place devices within the system. What will be interesting is how this technique is adapted to thermal maps, where the optimization criteria is not simply minima or maxima, but also regions of constant temperature, fluctuation maps, activity-based thermals, etc. Overall, the main research objective in all these

approaches is in the incorporation of thermal constraints in place of, or supplementing, VLSI-type constraints such as wire length and congestion.

8.1.2 Global Routing for Integrated Optics

Though we have presented effective techniques for detailed routing, we note that the global routing model presented in Chapter 5 is a rudimentary attempt at best. The presented MILP approach does, however, capture many of the important aspects required of an optical global router, notably how crossings and bends are modeled in a graph. What it lacks, however, is *scalability.*

Contemporary VLSI global routers have the ability to route problem instances many orders of magnitude more complex than the optical designs routed in our work. Though MILP-based approaches are utilized in VLSI routers, many are integrated into approaches that iteratively grow the routing regions to ensure tractability [46]. This iterative approach must also be adopted for our MILP-based approach to enable scalability. This will also benefit solution quality in that by reducing the number of nets under consideration, we can consider more candidate routes per net.

Finding good candidate routes is also an important part of creating a successful global router. In our MILP technique, candidate routes are selected based on the shortest routes available, which is justified by the relatively dense grid resulting from our row-based placement. In sparse placements with poorly defined grids, more traditional shape-based (e.g., Z or L) routing candidate nets may be more appropriate—provided they can still account for crossings and bends. With simpler shape-based MILP routing, a solution may not be found, requiring alternative approaches for routing. A crossing- and bend-aware maze router is therefore needed to supplement the other techniques, possibly even allowing for smoother bend transitions.

We also note that global routing and placement are increasingly becoming more integrated as they both affect each other. A coordinated place and route incorporating both the optical global routing as well as the placement of such devices may be the ultimate solution in optical network synthesis. Such an approach combines thermal placement with crossing- and bend-aware routing, ultimately to reduce power usage in the optical layer. For this to occur, the optical routing performed by such a framework needs to transition to a more iterative-friendly rather than constraint-solving methodology.

REFERENCES

[1] R. Soref, "The past, present, and future of silicon photonics," *IEEE J. Sel. Topics Quantum Electron.*, vol. 12, no. 6, pp. 1678–1687, 2006.

[2] C. Condrat, P. Kalla, and S. Blair, "Logic synthesis for integrated optics," in *Proc. 21st ACM Great Lakes Symp. VLSI*, ser. GLSVLSI '11. New York, NY, USA: ACM, 2011, pp. 13–18.

[3] ——, "Exploring design and synthesis for optical digital logic," in *Proc. Int. Workshop Logic Synth.*, 2010.

[4] H. J. Caulfield, C. S. Vikram, and A. Zavalin, "Optical logic redux," *Optik*, vol. 117, pp. 199–209, 2006.

[5] A. Politi, J. Matthews, and J. O'Brien, "Shor's quantum factoring algorithm on a photonic chip," *Science*, vol. 325, p. 1221, 2009.

[6] J. Hardy and J. Shamir, "Optics inspired logic architecture," *Opt. Express*, vol. 15, no. 1, pp. 150–165, 2007.

[7] H. J. Caulfield, R. A. Soref, L. Qian, A. Zavalin, and J. Hardy, "Generalized optical logic elements—GOLEs," *Opt. Commun.*, vol. 271, no. 2, pp. 365–376, 2007.

[8] P. Ganapati, "Germanium laser breakthrough brings optical computing closer," *Wired Mag.*, Feb. 2010.

[9] B. Steve and W. Kelvin, "Gated logic with optical solitons," in *Collision-based Computing*, A. Adamatzky, Ed. London, UK: Springer-Verlag, 2002, pp. 355–380.

[10] A. Shan, "Heterogeneous processing: A strategy for augmenting Moore's Law," *Linux J.*, vol. 2006, no. 142, p. 7, Feb. 2006.

[11] R. K. Dokania and A. B. Apsel, "Analysis of challenges for on-chip optical interconnects," in *Proc. 19th ACM Great Lakes Symp. VLSI*, ser. GLSVLSI '09. New York, NY, USA: ACM, 2009, pp. 275–280.

[12] C. Batten, A. Joshi, V. Stojanovic, and K. Asanovic, "Designing chip-level nanophotonic interconnection networks," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 2, no. 2, pp. 137–153, Jun. 2012.

[13] M. Cianchetti, J. Kerekes, and D. Albonesi, "Phastlane: A rapid transit optical routing network," in *Proc. 36th Int. Symp. Comput. Archit.*, ser. ISCA '09. New York, NY, USA: ACM, 2009, pp. 441–450.

[14] R. G. Beausoleil, J. Ahn, N. Binkert, A. Davis, D. Fattal, M. Fiorentino, N. P. Jouppi, M. McLaren, C. Santori, R. S. Schreiber, S. Spillane, D. Vantrease, and Q. Xu, "A nanophotonic interconnect for high-performance many-core computation," in *16th IEEE Symp. High Performance Interconnects*, Stanford, CA, Aug. 2008, pp. 182–189.

[15] J. Chan, G. Hendry, K. Bergman, and L. Carloni, "Physical-layer modeling and system-level design of chip-scale photonic interconnection networks," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 30, no. 10, pp. 1507–1520, Oct. 2011.

[16] S. J. Emelett and R. Soref, "Design and simulation of silicon microring optical routing switches," *J. Lightw. Technol.*, vol. 23, no. 4, p. 1800, 2005.

[17] H. Park, A. Fang, S. Kodama, and J. Bowers, "Hybrid silicon evanescent laser fabricated with a silicon waveguide and III-V offset quantum wells," *Opt. Express*, vol. 13, no. 23, pp. 9460–9464, 2005.

[18] J. Liu, X. Sun, R. Camacho-Aguilera, L. C. Kimerling, and J. Michel, "Ge-on-Si laser operating at room temperature," *Opt. Lett.*, vol. 35, no. 5, pp. 679–681, Mar. 2010.

[19] L. Liao *et al.*, "High speed silicon Mach–Zehnder modulator," *Opt. Express*, vol. 13, no. 8, pp. 3129–3135, 2005.

[20] W. Green *et al.*, "Ultra-compact, low RF power, 10 Gb/s silicon Mach–Zehnder modulator," *Opt. Express*, vol. 15, no. 25, pp. 17 106–17 113, 2007.

[21] M. J. R. Heck, H.-W. Chen, A. Fang, B. Koch, D. Liang, H. Park, M. Sysak, and J. Bowers, "Hybrid silicon photonics for optical interconnects," *IEEE J. Sel. Topics Quantum Electron.*, vol. 17, no. 2, pp. 333–346, Mar. 2011.

[22] "OpSIS: Optoelectronic System Integration in Silicon," http://www.opsisfoundry.org.

[23] C. Pina, "MOSIS: IC prototyping and low volume production service," in *Proc. Int. Conf. Microelectron. Syst. Education*, 2001, pp. 4–5.

[24] M. Peach, "Silicon photonics forges ahead," *Optics.org*, Feb. 2012.

[25] IBM Research, "Silicon Integrated Nanophotonics Technology: from the Lab to the Fab," http://www.research.ibm.com/photonics, 2012.

[26] Air Force Office of Scientific Research, "AFOSR: Complex Materials and Devices," 2013. [Online]. Available: http://www.wpafb.af.mil/library/factsheets/factsheet.asp?id=20370

[27] O. Boyraz and B. Jalali, "Demonstration of a silicon Raman laser," *Opt. Express*, vol. 12, no. 21, pp. 5269–5273, Oct. 2004.

[28] H. Rong, R. Jones, A. Liu, O. Cohen, D. Hak, A. Fang, and M. Paniccia, "A continuous-wave Raman silicon laser," *Nature*, vol. 433, pp. 725–728, 2005.

[29] R. W. Boyd, *Nonlinear Optics*, 3rd ed. Burlington, MA, USA: Academic Press, 2008.

[30] M. Dinu, F. Quochi, and H. Garcia, "Third-order nonlinearities in silicon at telecom wavelengths," *Appl. Phys. Lett.*, vol. 82, no. 18, pp. 2954–2956, 2003.

[31] M. R. Pearson *et al.*, "Arrayed waveguide grating demultiplexers in silicon-on-insulator," in *Proc. SPIE*, vol. 3953, 2000, pp. 11–18.

[32] A. P. Knights and J. K. Doylend, "Silicon photonics–recent advances in device development," *Adv. Inf. Opt. Photon.*, vol. PM183, pp. 633–656, Jun. 2008.

[33] G. Reed, W. Headley, and C. E. J. Png, "Silicon photonics: the early years," in *Proc. SPIE*, vol. 5730, 2005, pp. 1–18.

[34] R. A. Soref and B. Bennett, "Electrooptical effects in silicon," *IEEE J. Quantum Electron.*, vol. 23, no. 1, pp. 123–129, 1987.

[35] L. Liao, A. Liu, J. Basak, H. Nguyen, M. Paniccia, D. Rubin, Y. Chetrit, R. Cohen, and N. Izhaky, "40 Gbit/s silicon optical modulator for highspeed applications," *Electron. Lett.*, vol. 43, no. 22, 2007.

[36] M. Lipson, "Compact electro-optic modulators on a silicon chip," *IEEE J. Sel. Topics Quantum Electron.*, vol. 12, no. 6, pp. 1520 –1526, Nov. 2006.

[37] C. Gunn and G. L. I. Masini, "Closing in on photonics large-scale integration," *Photonics Spectra*, Dec. 2007.

[38] C. Pollock and M. Lipson, *Integrated Photonics*. Norwell, MA, USA: Kluwer Academic Publishers, 2003.

[39] D. A. B. Miller, "Optical interconnects to electronic chips," *Appl. Opt.*, vol. 49, no. 25, pp. F59–F70, Sep. 2010.

[40] C. Madsen and J. Zhao, *Optical Filter Design and Analysis: A Signal Processing Approach*, 1st ed. New York, NY, USA: John Wiley & Sons, Inc., 1999.

[41] *Lumerical Photonics CAD Suite*, Lumerical Solutions, Inc., http://www.lumerical.com.

[42] *RSoft Photonics CAD Suite*, RSoft Inc., http:://www.rsoftdesign.com.

[43] A. Mishchenko, S. Chatterjee, and R. Brayton, "DAG-aware AIG rewriting a fresh look at combinational logic synthesis," in *Proc. 43rd Design Autom. Conf.*, ser. DAC '06. New York, NY, USA: ACM, 2006, pp. 532–535.

[44] R. Bryant, "Graph-based algorithms for Boolean function manipulation," *IEEE Trans. Comput.*, vol. C-35, no. 8, pp. 677–691, Aug. 1986.

[45] A. B. Kahng, J. Lienig, I. L. Markov, and J. Hu, *VLSI Physical Design - From Graph Partitioning to Timing Closure.* Heidelberg, Germany: Springer, 2011.

[46] M. Cho, K. Lu, K. Yuan, and D. Pan, "BoxRouter 2.0: architecture and implementation of a hybrid and robust global router," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, Nov. 2007, pp. 503–508.

[47] M. Pan and C. Chu, "FastRoute 2.0: a high-quality and efficient global router," in *Proc. Asia and South Pacific Design Autom. Conf.*, 2007, pp. 250–255.

[48] A. Hashimoto and J. Stevens, "Wire routing by optimizing channel assignment within large apertures," in *Proc. 8th Design Autom. Workshop*, ser. DAC '71. New York, NY, USA: ACM, 1971, pp. 155–169.

[49] D. Deutsch, "A Dogleg channel router," in *Proc. 13th Design Autom. Conf.*, ser. DAC '76. New York, NY, USA: ACM, 1976, pp. 425–433.

[50] T. Yoshimura and E. Kuh, "Efficient algorithms for channel routing," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 1, no. 1, pp. 25–35, Jan. 1982.

[51] C. Condrat, P. Kalla, and S. Blair, "Channel routing for integrated optics," in *Proc. ACM/IEEE Int. Workshop Syst. Level Interconnect Prediction*, Austin, TX, Jun. 2013, pp. 1–8.

[52] ——, "Crossing-aware channel routing for photonic waveguides," in *Proc. IEEE Int. Midwest Symp. Circuits Syst.*, Aug. 2013, pp. 649–652.

[53] ——, "Thermal-aware synthesis of integrated photonic ring resonators," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, 2014.

[54] ——, "A methodology for physical design automation for integrated optics," in *Proc. IEEE Int. Midwest Symp. Circuits Syst.*, Aug. 2012, pp. 598–601.

[55] ——, "Crossing-aware channel routing for integrated optics," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 33, no. 6, pp. 814–825, Jun. 2014.

[56] K. Okamoto, *Fundamentals of Optical Waveguides*. Burlington, MA, USA: Academic Press, 2000.

[57] S. J. Koester *et al.*, "Ge-on-SOI-detector/Si-CMOS-amplifier receivers for high-performance optical-communication applications," *J. Lightw. Technol.*, vol. 25, no. 1, pp. 46–57, 2007.

[58] S. Emelett and R. Soref, "Analysis of dual-microring-resonator cross-connect switches and modulators," *Opt. Express*, vol. 13, no. 20, pp. 7840–7853, 2005.

[59] A. Sawchuk and T. Strand, "Digital optical computing," *Proc. IEEE*, vol. 72, pp. 758–779, 1984.

[60] D. E. Chang, A. S. Sørensen, E. A. Demler, and M. D. Lukin, "A single-photon transistor using nanoscale surface plasmons," *Nature Physics*, Aug. 2007.

[61] C. Wenlan, K. M. Beck, B. Robert, G. Michael, M. D. Lukin, T. Haruka, and V. Vladan, "All-optical switch and transistor gated by one stored photon," *Science*, 2013.

[62] A. T. Pham and L. N. Binh, "All-optical modulation and switching using a nonlinear-optical directional coupler," *J. Opt. Soc. Am. B*, vol. 8, no. 9, pp. 1914–1931, Sep. 1991.

[63] Y. Wang and J. Liu, "All-fiber logical devices based on the nonlinear directional coupler," *IEEE Photon. Technol. Lett.*, vol. 11, no. 1, pp. 72–74, 1999.

[64] J. P. Sokoloff, P. Prucnal, I. Glesk, and M. Kane, "A terahertz optical asymmetric demultiplexer (TOAD)," *IEEE Photon. Technol. Lett.*, vol. 5, no. 7, pp. 787–790, 1993.

[65] N. J. Doran and D. Wood, "Nonlinear-optical loop mirror," *Opt. Lett.*, vol. 13, no. 1, pp. 56–58, Jan. 1988.

[66] A. Huang, N. Whitaker, H. Avramopoulos, P. French, H. Houh, and I. Chuang, "Sagnac fiber logic gates and their possible applications: a system perspective," *Appl. Opt.*, vol. 33, no. 26, pp. 6254–6267, Sep. 1994.

[67] C. Koos, L. Jacome, C. Poulton, J. Leuthold, and W. Freude, "Nonlinear silicon-on-insulator waveguides for all-optical signal processing," *Opt. Express*, vol. 15, no. 10, pp. 5976–5990, May 2007.

[68] J. Zhang, Q. Lin, G. Piredda, R. W. Boyd, G. P. Agrawal, and P. M. Fauchet, "Optical solitons in a silicon waveguide," *Opt. Express*, vol. 15, no. 12, pp. 7682–7688, Jun. 2007.

[69] A. J. Poustie and K. J. Blow, "Demonstration of an all-optical Fredkin gate," *Opt. Commun.*, vol. 174, no. 1-4, pp. 317–320, 2000.

[70] J. Shamir, H. J. Caulfield, W. Micelli, and R. J. Seymour, "Optical computing and the Fredkin gates," *Appl. Opt.*, vol. 25, no. 10, pp. 1604–1607, 1986.

[71] D. Ding, Y. Zhang, H. Huang, R. T. Chen, and D. Z. Pan, "O-Router: an optical routing framework for low power on-chip silicon nano-photonic integration," in *Proc. 46th Design Autom. Conf.*, ser. DAC '09. New York, NY, USA: ACM, 2009, pp. 264–269.

[72] K. Shlager and J. Schneider, "A selective survey of the finite-difference time-domain literature," *IEEE Antennas Propag. Mag.*, vol. 37, no. 4, pp. 39–57, Aug. 1995.

[73] A. Mishchenko and M. Perkowski, "Fast heuristic minimization of exclusive-sums-of-products," in *Proc. Reed–Muller Workshop*, 2001, pp. 242–250.

[74] T. Sasao, "An exact minimization of AND-EXOR expressions using BDDs," in *Proc. IFIP 10.5 Workshop Appl. Reed-Muller Expansion in Circuit Design*, Sep. 1993.

[75] C. Yang and M. Ciesielski, "BDS: a BDD-based logic optimization system," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 21, no. 7, pp. 866–876, Jul. 2002.

[76] T. S. Czajkowski and S. D. Brown, "Functionally linear decomposition and synthesis of logic circuits for FPGAs," in *Proc. 45th Design Autom. Conf.*, ser. DAC '08. New York, NY, USA: ACM, 2008, pp. 18–23.

[77] P. Gupta, A. Agrawal, and N. K. Jha, "An algorithm for synthesis of reversible logic circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 25, no. 11, pp. 2317–2330, 2006.

[78] R. Wille and R. Drechsler, "BDD-based synthesis of reversible logic for large functions," in *Proc. 46th Design Autom. Conf.*, Jul. 2009, pp. 270–275.

[79] ACM/SIGDA, "ACM/SIGDA benchmarks," http://www.cbl.ncsu.edu/benchmarks/Benchmarks-upto-1996.html, 1996.

[80] D. Ding and D. Pan, "OIL: A nano-photonics optical interconnect library for a new photonic networks-on-chip architecture," in *Proc. ACM/IEEE Int. Workshop Syst. Level Interconnect Prediction*, ser. SLIP '09. New York, NY, USA: ACM, 2009, pp. 11–18.

[81] J. Orcutt and R. Ram, "Photonic device layout within the foundry CMOS design environment," *IEEE Photon. Technol. Lett.*, vol. 22, no. 8, pp. 544–546, Apr. 2010.

[82] D. Ding, B. Yu, and D. Pan, "GLOW: A global router for low-power thermal-reliable interconnect synthesis using photonic wavelength multiplexing," in *Proc. Asia and South Pacific Design Autom. Conf.*, Feb. 2012, pp. 621–626.

[83] Y. Zheng, P. Lisherness, M. Gao, J. Bovington, K. Cheng, H. Wang, and S. Yang, "Power-efficient calibration and reconfiguration for optical network-on-chip," *J. Opt. Commun. Netw.*, vol. 4, no. 12, pp. 955–966, Dec. 2012.

[84] A. Boos, L. Ramini, U. Schlichtmann, and D. Bertozzi, "PROTON: An automatic place-and-route tool for optical networks-on-chip," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, 013, pp. 138–145.

[85] S. Kotiyal, H. Thapliyal, and N. Ranganathan, "Mach–Zehnder interferometer based design of all optical reversible binary adder," in *Proc. Design, Automation, and Test in Europe*, Mar. 2012, pp. 721–726.

[86] W. Bogaerts, P. Dumon, D. V. Thourhout, and R. Baets, "Low-loss, low-cross-talk crossings for silicon-on-insulator nanophotonic waveguides," *Opt. Lett.*, vol. 32, no. 19, pp. 2801–2803, Oct. 2007.

[87] P. Sanchis, P. Villalba, F. Cuesta, A. Håkansson, A. Griol, J. V. Galán, A. Brimont, and J. Martí, "Highly efficient crossing structure for silicon-on-insulator waveguides," *Opt. Lett.*, vol. 34, no. 18, pp. 2760–2762, Sep. 2009.

[88] F. Xu and A. W. Poon, "Silicon cross-connect filters using microring resonator coupled multimode-interference-based waveguide crossings," *Opt. Express*, vol. 16, no. 12, pp. 8649–8657, Jun. 2008.

[89] J. Cardenas, C. B. Poitras, J. T. Robinson, K. Preston, L. Chen, and M. Lipson, "Low loss etchless silicon photonic waveguides," *Opt. Express*, vol. 17, no. 6, pp. 4752–4757, Mar. 2009.

[90] Y. Vlasov and S. McNab, "Losses in single-mode silicon-on-insulator strip waveguides and bends," *Opt. Express*, vol. 12, no. 8, pp. 1622–1631, Apr. 2004.

[91] Y. Qian, S. Kim, J. Song, G. P. Nordin, and J. Jiang, "Compact and low loss silicon-on-insulator rib waveguide 90° bend," *Opt. Express*, vol. 14, no. 13, pp. 6020–6028, Jun. 2006.

[92] G. Li, J. Yao, H. Thacker, A. Mekis, X. Zheng, I. Shubin, Y. Luo, J. Lee, K. Raj, J. E. Cunningham, and A. V. Krishnamoorthy, "Ultralow-loss, high-density SOI optical waveguide routing for macrochip interconnects," *Opt. Express*, vol. 20, no. 11, pp. 12 035–12 039, May 2012.

[93] J. Roy, D. Papa, S. Adya, H. Chan, A. Ng, J. Lu, and I. Markov, "Capo: robust and scalable open-source min-cut floorplacer," in *Proc. Int. Symp. Physical Design*, ser. ISPD '05. New York, NY, USA: ACM, 2005, pp. 224–226.

[94] M. Larry and E. Carl, "PathFinder: A Negotiation-based Performance-driven Router for FPGAs," in *Proc. 3rd Int. Symp. Field-Programmable Gate Arrays*, ser. FPGA '95. New York, NY, USA: ACM, 1995, pp. 111–117.

[95] Y. Chang, Y. Lee, and T. Wang, "NTHU-Route 2.0: A fast and stable global router," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, 2008, pp. 338–343.

[96] T. Lee and T. Wang, "Congestion-constrained layer assignment for via minimization in global routing," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 27, no. 9, pp. 1643–1656, 2008.

[97] E. de Queiros Vieira Martins and M. Pascoal, "A new implementation of Yen's ranking loopless paths algorithm," *Quart. J. Belgian, French, and Italian Oper. Res. Soc.*, vol. 1, no. 2, pp. 121–133, 2003.

[98] G. O. Inc., "Gurobi optimizer reference manual," 2013. [Online]. Available: http://www.gurobi.com

[99] J. Yao, X. Zheng, G. Li, I. Shubin, H. Thacker, Y. Luo, K. Raj, J. Cunningham, and A. Krishnamoorthy, "Grating-coupler based low-loss optical interlayer coupling," in *Proc. 8th IEEE Int. Conf. Group IV Photon.*, Sep. 2011, pp. 383 –385.

[100] X. Zheng *et al.*, "Optical proximity communication using reflective mirrors," *Opt. Express*, vol. 16, no. 19, pp. 15 052–15 058, Sep. 2008.

[101] K. Chaudhary and P. Robinson, "Channel routing by sorting," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 10, no. 6, pp. 754–760, Jun. 1991.

[102] J. Yan, "An improved optimal algorithm for bubble-sorting-based non-Manhattan channel routing," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 18, no. 2, pp. 163–171, Feb. 1999.

[103] S. Sait and H. Youssef, *VLSI Physical Design Automation: Theory and Practice*. New York City, NY, USA: McGraw-Hill, 1995.

[104] M. Marek-Sadowska and M. Sarrafzadeh, "The crossing distribution problem [IC layout]," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 14, no. 4, pp. 423–433, Apr.

[105] S. Xiaoyu and W. Yuke, "On the crossing distribution problem," *ACM Trans. Design Autom. Electron. Syst.*, vol. 4, no. 1, pp. 39–51, Jan. 1999.

[106] S. B. Stephen and L. S. Kyu, "QCA channel routing with wire crossing minimization," in *Proc. 15th ACM Great Lakes Symp. VLSI*, ser. GLSVLSI '05. New York, NY, USA: ACM, 2005, pp. 217–220.

[107] C. Cheng and D. N. Deutsch, "Improved channel routing by via minimization and shifting," in *Proc. 25th Design Autom. Conf.*, 1988, pp. 677–680.

[108] G. Nam, C. Sze, and M. Yildiz, "The ISPD global routing benchmark suite," in *Proc. Int. Symp. Physical Design*, ser. ISPD '08. New York, NY, USA: ACM, 2008, pp. 156–159.

[109] E. A. J. Marcatili and S. E. Miller, "Improved relationships describing directional control in electromagnetic wave guidance," *Bell Syst. Tech J.*, vol. 48, pp. 2161–2188, 1969.

[110] W. Bogaerts *et al.*, "Silicon microring resonators," *Laser & Photonics Reviews*, vol. 6, no. 1, pp. 47–73, 2012.

[111] N. Rouger, L. Chrostowski, and R. Vafaei, "Temperature effects on silicon-on-insulator (SOI) racetrack resonators: a coupled analytic and 2D finite difference approach," *J. Lightw. Technol.*, vol. 28, pp. 1380–1391, May 2010.

[112] F. Gan, T. Barwicz, M. Popovic, M. Dahlem, C. Holzwarth, P. T. Rakich, H. Smith, E. Ippen, and F. Kartner, "Maximizing the thermo-optic tuning range of silicon photonic structures," in *Photonics in Switching*, 2007, pp. 67–68.

[113] C. Shih, Z. Zeng, and C. Shiuh, "Extinction ratio compensation by free carrier injection for a MOS-capacitor microring optical modulator subjected to temperature drifting," in *Proc. Conf. Lasers Electro Optics / Pacific Rim Conf. Lasers and Electro-Optics*, Aug. 2009, pp. 1–2.

[114] J. Schrauwen, D. Thourhout, and R. Baets, "Trimming of silicon ring resonator by electron beam induced compaction and strain," *Opt. Express*, vol. 16, no. 6, pp. 3738–3743, Mar. 2008.

[115] S. Lambert, W. D. Cort, J. Beeckman, K. Neyts, and R. Baets, "Trimming of silicon-on-insulator ring resonators with a polymerizable liquid crystal cladding," *Opt. Lett.*, vol. 37, no. 9, pp. 1475–1477, May 2012.

[116] J. Teng, P. Dumon, W. Bogaerts, H. Zhang, X. Jian, M. Zhao, G. Morthier, and R. Baets, "Athermal SOI ring resonators by overlaying a polymer cladding on narrowed waveguides," in *Proc. 6th IEEE Int. Conf. Group IV Photon.*, 2009, pp. 77–79.

[117] K. Hill and G. Meltz, "Fiber Bragg grating technology fundamentals and overview," *J. Lightw. Technol.*, vol. 15, no. 8, pp. 1263–1276, 1997.

[118] E. Dulkeith, F. Xia, L. Schares, W. M. J. Green, and Y. A. Vlasov, "Group index and group velocity dispersion in silicon-on-insulator photonic wires," *Opt. Express*, vol. 14, no. 9, pp. 3853–3863, May 2006.

[119] M. Kim, D. Lee, and I. Markov, "SimPL: An effective placement algorithm," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 31, no. 1, pp. 50–60, Jan. 2012.

[120] J. Hu, M. Kim, and I. Markov, "Taming the complexity of coordinated place and route," in *Proc. 50th Design Autom. Conf.*, 2013, pp. 1–7.